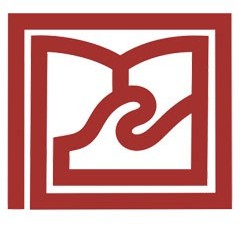
**ĐẠI HỌC DUY TÂN**

**TRƯỜNG KHOA HỌC MÁY TÍNH**

**KHOA KỸ THUẬT MẠNG MÁY TÍNH & TRUYỀN THÔNG**





**ĐỒ ÁN NHÓM:**

**MÔN: PERL & PYTHON – MÃ MÔN: CS466**

**Tên đề tài:**

**LẬP TRÌNH WEBSITE MẠNG XÃ HỘI**

**BẰNG DJANGO TRONG PYTHON**

|  |  |
| --- | --- |
| Nhóm thực hiện: Nhóm 18  Sinh viên 1: Ngô Nguyễn Thùy Linh  MSSV: 28201139984  Sinh viên 2: Trần Thị Thanh Giang  MSSV: 28201154741  Lớp môn học: CS 466 E  GVHD: ThS. Trần Hữu Minh Đăng | |
| *Đà Nẵng, ngày 18 tháng 12 năm 2024* | |

**MỤC LỤC:**

[MỞ ĐẦU 5](#_Toc185379605)

[A. Lý do chọn tài liệu 5](#_Toc185379606)

[B. Mục tiêu 5](#_Toc185379607)

[C. Object, phạm vi nghiên cứu 6](#_Toc185379608)

[D. Phương pháp nghiên cứu 7](#_Toc185379609)

[E. Nhóm sơ đồ cấu hình 7](#_Toc185379610)

[Kết luận: 7](#_Toc185379611)

[CHƯƠNG 1: TỔNG QUAN VỀ PYTHON 8](#_Toc185379612)

[I. Giới thiệu chung 8](#_Toc185379613)

[1. Lịch Sử Phát Triển 8](#_Toc185379614)

[2. Đặc Điểm Nổi Bật của Python 8](#_Toc185379615)

[II. Ngôn Ngữ Lập Trình Python 8](#_Toc185379616)

[1. Cú Pháp và Cấu Trúc Ngôn Ngữ 8](#_Toc185379617)

[2. Thư Viện, Môđun và khung 12](#_Toc185379618)

[3. Ưu và Nhược Điểm 13](#_Toc185379619)

[4. Các Ứng Dụng Thực Tế 14](#_Toc185379620)

[CHƯƠNG 2: PHÂN TÍCH & THIẾT KẾ HỆ THỐNG BẰNG DJANGO TRONG PYTHON 15](#_Toc185379621)

[I. Phân tích yêu cầu hệ thống 15](#_Toc185379622)

[1. Yêu cầu chức năng 15](#_Toc185379623)

[2. Yêu cầu phi chức năng 15](#_Toc185379624)

[II. Thiết kế kiến trúc hệ thống 16](#_Toc185379625)

[1. Kiến trúc tổng thể 16](#_Toc185379626)

[III. Sơ đồ kiến trúc hệ thống 17](#_Toc185379627)

[IV. Sơ đồ luồng dữ liệu 18](#_Toc185379628)

[V. Thiết kế giao diện người dùng (UI) 19](#_Toc185379629)

[1. Nguyên tắc thiết kế 19](#_Toc185379630)

[2. Mẫu giao diện 20](#_Toc185379631)

[VI. Thiết kế cơ sở dữ liệu 24](#_Toc185379632)

[1. Mô hình dữ liệu 24](#_Toc185379633)

[2. Sơ đồ thiết kế cơ sở dữ liệu 24](#_Toc185379634)

[VII. Kết luận chương 2 26](#_Toc185379635)

[CHƯƠNG 3: TRIỂN KHAI VÀ KIỂM THỬ HỆ THỐNG 27](#_Toc185379636)

[I. Công cụ sử dụng 27](#_Toc185379637)

[II. Triển khai hệ thống 28](#_Toc185379638)

[1. Kịch bản thứ 1: Mô-đun Đăng Ký Người Dùng 28](#_Toc185379639)

[2. Kịch bản 2: Mô-đun Đăng Nhập Người Dùng (User Login) 29](#_Toc185379640)

[3. Kịch bản 3: Cấu hình account settings 30](#_Toc185379641)

[4. Kịch bản 4: Cấu hình trang chủ 31](#_Toc185379642)

[5. Kịch bản 5: Cấu hình trang cá nhân 34](#_Toc185379643)

[III. Kiểm thử hệ thống 36](#_Toc185379644)

[1. Các loại kiểm thử 36](#_Toc185379645)

[2. Kế hoạch kiểm thử 37](#_Toc185379646)

[3. Kết quả đạt được 37](#_Toc185379647)

[IV. KẾT LUẬN 37](#_Toc185379648)

[1. Kết quả đạt được 37](#_Toc185379649)

[2. Hạn chế: 37](#_Toc185379650)

[3. Kết luận chương 3 38](#_Toc185379651)

[KẾT LUẬN 39](#_Toc185379652)

[TÀI LIỆU THAM KHẢO 40](#_Toc185379653)

# MỞ ĐẦU

## Lý do chọn tài liệu

Lý do chọn đề tài Nhóm Chọn đề tài về trang mạng xã hội:

- Tính thực cao: Nền tảng xã hội mang lại nhiều lợi ích như giao tiếp, chia sẻ thông tin, giải trí và thậm chí hỗ trợ kinh doanh chính yếu

- Đóng góp xã hội: Tạo môi trường trải nghiệm người dùng lành mạnh giúp kết nối người dùng với nhau qua các bài viết chia sẻ về trải nghiệm, kinh nghiệm và thắc mắc về nhiều lĩnh vực. Đồng thời giúp đỡ những người không còn khả năng lao động mạnh, mẹ tắmm sữa chia sẻ và đăng bài viết bán hàng kiếm thêm thu nhập

## Mục tiêu

1. Tìm hiểu và thiết kế phát triển kiến ​​trúc trang web

Xây dựng cấu hình ứng dụng

- Sử dụng mô hình MVC (Model-View-Controller)

- Tổ chức dự án thành các ứng dụng thu gọn, dễ quản lý như: người dùng, bài viết,thông báo.

Quản lý URL: Sử dụng urls.py

2. Hiểu và sử dụng các framework Django, RESTful API trong Python để phát triển sử dụng web

Django: Nghiên cứu cách sử dụng các framework này để phát triển ứng dụng web.

API: Tìm hiểu cách xây dựng API để phục vụ việc bán ứng dụng.

3. Xử lý dữ liệu và cơ sở dữ liệu

Thiết kế cơ sở dữ liệu:

- Sử dụng Django ORM để định nghĩa các Model đại diện cho

- Liên kết dữ liệu với các mối quan hệ: One-to-One, Một-nhiều , Nhiều-nhiều

Quản lý dữ liệu động:

- Tạo, đọc, cập nhật và xóa dữ liệu (CRUD) một cách dễ dàng thông qua thông tin ORM.

- Sử dụng lệnh makemigrations và Migration để đồng bộ hóa mô hình với cơ sở dữ liệu.

4. Xây dựng giao diện người dùng (UI)

Use Django Sample

- Kết hợp HTML với các biến (mẫu) và logic điều kiện (if, for) để hiển thị nội dung.

Giao diện thiết kế:

- Tích hợp framework CSS như Bootstrap, Tailwind CSS để tạo giao diện đẹp mắt

- Tạo các trang chính: Trang chủ, trang đăng nhập, đăng ký, danh sách bài viết, chi tiết

Quản lý trang web: Thêm, sửa, xóa trang web .

5. Quản lý người dùng

Xác thực hệ thống :

- Cung cấp tính năng đăng ký, đăng nhập, đăng xuất dựa trên hệ thống xác thực của Django.

Bảo vệ dữ liệu:

- Áp dụng các biện pháp bảo mật để bảo vệ thông tin người dùng.

- Bảo vệ mật khẩu bằng cách sử dụng hàm băm hợp lý.

Quản lý tài khoản người dùng:

- Hỗ trợ chức năng đổi mật khẩu, khôi phục mật khẩu qua email.

- Xây dựng hồ sơ người dùng (hồ sơ) các trang để hiển thị thông tin cá nhân.

6. Quản lý và bảo mật dữ liệu

Quyền truy cập

- Kiểm soát quyền truy cập (quyền) dựa trên loại người dùng (quản trị viên, người dùng).

Bảo mật

- Kích hoạt bảo vệ CSRF để ngăn chặn hành vi giả mạo yêu cầu xuyên trang web.

- Sử dụng tính năng bảo vệ mặc định của Django để chống SQL Insert, XSS.

7. Xử lý logic và chức năng

Tạo quá trình xử lý logic trong view.py

- Xây dựng chế độ xem dựa trên lớp (CBV) hoặc chế độ xem dựa trên chức năng (FBV) để xử lý yêu cầu của người dùng.

Ví dụ: Hiển thị danh sách bài viết, chi tiết bài viết, thêm bài viết mới.

Sử dụng Django Signals : Tự động kích hoạt khi có sự kiện xảy ra, ví dụ: Gửi email khi người dùng mới đăng ký.

8. Triển khai và tối ưu

Khai báo trang web : Đưa trang web lên các nền tảng như Heroku, AWS hoặc VPS Cấu hình Gunicorn hoặc Nginx để chạy ứng dụng trong môi trường thực thi.

Hiệu suất tối ưu :

- Tích hợp bộ nhớ đệm để cải thiện tốc độ tải trang.

- Tối ưu truy vấn cơ sở dữ liệu với select\_similar và prefetch\_similar.

9. Các tính năng nâng cao

Ứng dụng cài đặt giao diện:

- Tích hợp Django Rest Framework (DRF) để xây dựng API dịch vụ di động ứng dụng và các ứng dụng khác .

- Tính năng thời gian thực : Sử dụng Django Channels để hỗ trợ các tính năng thời gian thực như trò chuyện trực tuyến hoặc thông báo

## Object, phạm vi nghiên cứu

Đối tượng:

- Mạng xã hội website: Nghiên cứu và phát triển một trang web ứng dụng cho phép người dùng tìm kiếm, xem, giao tiếp, chia sẻ thông tin, giải quyết, hỗ trợ kinh doanh - Đóng góp xã hội.

- Người dùng: Người dùng đối tượng, bao gồm những người có sở hữu doanh nghiệp, nhu cầu học hỏi cao hoặc giải trí, sẽ tương tác với trang web.

- Công nghệ sử dụng: Tập trung vào trình cài đặt công nghệ Python, cùng với các framework Django và các công cụ hỗ trợ như cơ sở dữ liệu (PostgreSQL, MySQL) và HTTP giao thức.

Phạm vi:

- Chức năng chính: Xây dựng các cơ sở chức năng của trang web xã hội giúp người dùng tương tác với nhau và kinh doanh trên trang .

- Thời gian: Nghiên cứu và phát triển trong một khoảng thời gian cụ thể, ví dụ, trong 3-6 tháng.

- Đối tượng người dùng: những người có nhu cầu giải trí, kinh doanh không giới hạn tuổi và sự tương tác với người dùng.

## Phương pháp nghiên cứu

Lý thuyết:

- Nghiên cứu tài liệu: Tìm hiểu các trang mạng khác nhau về python , đọc và học hỏi những điều chưa biết và cần biết để có thể hiểu thêm về sự thiết lập trang web về trang mạng xã hội trong python.

- Khung lý thuyết: Xây dựng khung lý thuyết về quy tắc cài đặt trang web, thiết kế giao diện người dùng (UI) và trải nghiệm người dùng (UX) để đảm bảo sản phẩm cuối cùng đáp ứng nhu cầu của ngư dân tôi sử dụng.

- Phân tích công nghệ: Nghiên cứu và tìm hiểu thêm về trang web hiện tại đang có ở django trong python và cần hiểu chiều sâu và các công nghệ cũng như các ứng dụng mà django đạt được trong python.

Thực nghiệm:

Xây dựng nguyên mẫu: Thiết kế giao diện phù hợp với người dùng thoải mái, tạo cảm giác thoải mái, tối ưu nhất cho mọi người đặt biệt là những đối tượng trên trang mạng xã hội.

Kiểm tra chức năng: Tiến hành kiểm tra các chức năng của trang web như bổ sung chỉnh sửa các bài đăng theo cách phù hợp, cho phép đăng ảnh và chia sẻ hình ảnh những nội dung phù hợp và có những trang cần đánh giá vi phạm cộng đồng đối đầu với các đối tượng gây nguy hiểm cho trang mạng xã hội của trang web .

Triển khai thử nghiệm: Đưa ứng dụng lên môi trường thực tế trong một khoảng thời gian ngắn để theo dõi hành vi của người dùng và đánh giá giá của người dùng về trang web như thế nào và có những giải pháp tối ưu nhất đối với phản hồi không tốt từ sử dụng.

## Nhóm sơ đồ cấu hình

Đồ án bao gồm phần mở đầu, ba chương và phần thảo luận.

Chương 1: Tổng quan về website trang mạng xã hội

Chương 2: Tìm hiểu về Tornado công cụ phân tích và thiết kế mạng xã hội trang web.

Chương 3: Triển khai & kiểm tra hệ thống

## Kết luận:

Tóm tắt những kết quả đạt được trong quá trình thực thi cũng như sử dụng sản phẩm như thế nào về trang web trang mạng xã hội .Đánh giá những công thức khó khăn trong quá trình xây dựng trang web cũng như trong quá trình trải nghiệm, rút ra được bài học, kinh nghiệm từ những dự án tương tự. Cũng giống như những bức phá trong quá trình phát triển website để trong tương lai được mở rộng thêm các cức năng mới mà người dùng mong muốn và hơn thế nữa, cải trải nghiệm của người dùng được cảithiện, áp dụng được sử dụng các công nghệ mới và triển khai vào mạng xã hội của trang web.

# CHƯƠNG 1: TỔNG QUAN VỀ PYTHON

### **Giới thiệu chung**

Python là một ngôn ngữ lập trình thông dịch, được sử dụng đa mục đích, nổi tiếng với cú pháp đơn giản và dễ đọc, và cho phép người lập trình viết mã một cách tự nhiên. Python có một hệ thống thư viện phong phú và mạnh mẽ, giúp giải quyết nhiều loại vấn đề khác nhau. Vì vậy, ngôn ngữ này thường được sử dụng trong phát triển ứng dụng web, khoa học dữ liệu, trí tuệ nhân tạo và nhiều lĩnh vực khác.

#### Lịch Sử Phát Triển

Python được phát triển vào cuối những năm 1980 bởi Guido van Rossum và phát hành lần đầu vào năm 1991. Lấy cảm hứng từ ngôn ngữ ABC và chương trình hài *Monty Python's Flying Circus*, Python hướng đến sự dễ sử dụng và thân thiện.

* **Python 1.0** (1994): Ra mắt với các tính năng cơ bản như list, str, và công cụ xử lý dữ liệu.
* **Python 2.0** (2000): Thêm thu gom rác tự động, hỗ trợ Unicode. Python 2.7 là phiên bản cuối, được hỗ trợ đến năm 2020.
* **Python 3.0** (2008): Cải tiến lớn, không tương thích ngược với Python 2, tập trung vào việc đơn giản hóa cú pháp và cải thiện hiệu năng.

Python đã phát triển mạnh mẽ, trở thành ngôn ngữ phổ biến nhờ cú pháp rõ ràng, thư viện phong phú và ứng dụng rộng rãi trong phát triển web, khoa học dữ liệu, AI, và tự động hóa.

**Python**: Ngôn ngữ đa năng và dễ học, phổ biến trong phát triển web, AI, khoa học dữ liệu, và các ứng dụng hiện đại khác. Python có một cộng đồng lớn và nhiều thư viện hỗ trợ cho hầu hết mọi lĩnh vực.

#### Đặc Điểm Nổi Bật của Python

* **Dễ Học và Dễ Sử Dụng**: Cú pháp của Python gần gũi với ngôn ngữ tự nhiên, giúp cho việc học và phát triển các chương trình dễ dàng hơn ngay cả đối với người mới bắt đầu.
* **Mã Nguồn Mở và Miễn Phí**: Python là ngôn ngữ mã nguồn mở, được phát hành dưới giấy phép GNU General Public License (GPL), cho phép sử dụng, phân phối và phát triển miễn phí.
* **Đa Năng**: Python là ngôn ngữ đa năng, có thể sử dụng trong nhiều lĩnh vực khác nhau, từ phát triển web, tự động hóa, khoa học dữ liệu, trí tuệ nhân tạo, đến lập trình hệ thống.
* **Thư Viện Phong Phú**: Python có một hệ sinh thái thư viện đồ sộ và một cộng đồng hỗ trợ rộng lớn, giúp giải quyết hầu hết các vấn đề lập trình. Các thư viện nổi bật bao gồm Pandas, NumPy, Matplotlib, TensorFlow, và Flask.
* **Tính Mô-đun và Hướng Đối Tượng**: Python hỗ trợ lập trình mô-đun và hướng đối tượng, cho phép tổ chức mã nguồn một cách dễ dàng và hiệu quả.
* **Quản Lý Bộ Nhớ Tự Động**: Python có cơ chế quản lý bộ nhớ tự động thông qua hệ thống thu gom rác (garbage collection), giúp giảm thiểu rủi ro rò rỉ bộ nhớ.

### **Ngôn Ngữ Lập Trình Python**

#### Cú Pháp và Cấu Trúc Ngôn Ngữ

* 1. Biến và Kiểu Dữ Liệu
     1. Biến

Biến là nơi chứa các giá trị dữ liệu.

|  |
| --- |
| Cách đặt tên biến: |

Một biến có thể có tên ngắn (như a và b) hoặc tên mang tính mô tả hơn (name, age and sum). Quy tắc cho các biến Python:

* Tên biến phải bắt đầu bằng một chữ cái hoặc ký tự gạch dưới
* Tên biến không thể bắt đầu bằng số
* Tên biến chỉ có thể chứa các ký tự chữ và số và dấu gạch dưới
* Tên biến có phân biệt chữ hoa chữ thường (ten, name và Name là ba biến khác nhau)
* Tên biến không thể là bất kỳ từ khóa Python (False, None, True, and, as, break, class, continue, def, …)
  + 1. Kiểu dữ liệu

Python là một ngôn ngữ lập trình có kiểu dữ liệu động, tức là bạn không cần phải xác định kiểu dữ liệu của biến mà Python sẽ tự động xác định nó dựa trên giá trị mà bạn gán cho biến. Dưới đây là một số kiểu dữ liệu phổ biến trong Python:

**Kiểu Integer (int):** Kiểu số nguyên (không có chứa số thập phân), có thể lưu các số nguyên âm và dương.

Ví dụ: number = 12345

**Kiểu Float (float):** Được sử dụng để biểu diễn số thập phân.

Ví dụ: number = 3.14

**Kiểu Complex:** Được sử dụng để biểu diễn số phức với một phần thực và một

phần ảo.

Ví dụ 1: z = 2+3j thì 2 là phần thực, 3 là phần ảo (j là từ khóa để đánh dấu phần ảo)

**Kiểu None:** Được sử dụng để biểu diễn giá trị null hoặc không tồn tại.

**Kiểu DateTime:** Được sử dụng để làm việc với thời gian và ngày tháng.

**Kiểu Bytes và Bytearray:** Được sử dụng để làm việc với dữ liệu nhị phân.

**Kiểu Range:** Được sử dụng để biểu diễn một phạm vi số nguyên tuần tự.

Ví dụ: range(10)

**Kiểu Boolean (bool):** Được sử dụng để biểu diễn giá trị True hoặc False.

Ví dụ: True, False.

**Kiểu String (str):** Kiểu chuỗi, để trong nháy đôi hoặc nháy đơn

Ví dụ: "Chào", 'Python and Perl'.

**Kiểu List:** Là một cấu trúc dữ liệu dạng danh sách có thể chứa nhiều giá trị

giống nhau hoặc khác nhau được phân tách bằng dấu phẩy và được đặt trong dấu ngoặc vuông [ ]

Ví dụ: [1, 2, 3, 4, 5]; ['python', 'and', 'perl'].

**Kiểu Tuple:** Tương tự như kiểu List, nhưng không thể thay đổi sau khi được

tạo. Trong Python, chúng ta sử dụng dấu ngoặc đơn () để lưu trữ các mục của một tuple

**Kiểu Set :** Là một cấu trúc dữ liệu không có thứ tự và không chứa các phần

tử trùng lặp. Set được định nghĩa bởi các giá trị được phân tách bằng dấu phẩy bên trong dấu ngoặc nhọn { }

**Kiểu Dictionary :** là một tập hợp các mục được sắp xếp. Nó lưu trữ các phần tử theo cặp khóa/giá trị. Ở đây, khóa là các mã định danh duy nhất được liên kết với mỗi giá trị.

* 1. Điều Khiển Luồng

Cấu trúc điều khiển trong Python là các thành phần giúp điều khiển luồng thực thi của chương trình. Chúng cho phép chương trình thực hiện các hành động khác nhau tùy thuộc vào các điều kiện cụ thể. Cấu trúc điều khiển phổ biến bao gồm if, if-else, if-elif-else, và vòng lặp for, while.

* 1. Cấu trúc if trong python

Lệnh if trong Python là giống như trong ngôn ngữ C. Lệnh này được sử dụng để kiểm tra một điều kiện, nếu điều kiện là true thì lệnh của khối if sẽ được thực thi, nếu không nó sẽ bị bỏ qua.

Cú pháp của lệnh if là:

if bieu\_thuc:

cac\_lenh

* + 1. Cấu trúc if-else trong python

Một lệnh else có thể được sử dụng kết hợp với lệnh if. Một lệnh else chứa khối code mà thực thi nếu biểu thức điều kiện trong lệnh if được ước lượng là 0 hoặc một giá trị false. Lệnh else là lệnh tùy ý và chỉ có duy nhất một lệnh else sau lệnh if.

Cú pháp của lệnh if...else là:

if bieu\_thuc:

cac\_lenh

else:

cac\_lenh

* + 1. Cấu trúc if lồng nhau trong python

Lệnh elif cho phép bạn kiểm tra nhiều điều kiện và thực thi khối code ngay khi một trong các điều kiện được ước lượng là true. Cũng giống như lệnh else, lệnh elif là tùy ý. Tuy nhiên, không giống else mà chỉ có một lệnh được theo sau if, thì bạn có thể sử dụng **nhiều elif theo sau if**.

Cú pháp của lệnh elif là:

if bieu\_thuc1:

   cac\_lenh

elif bieu\_thuc2:

   cac\_lenh

elif bieu\_thuc3:

   cac\_lenh

else:

   cac\_lenh

Python không cung cấp các lệnh switch hoặc case như trong các ngôn ngữ lập trình khác, tuy nhiên bạn có thể sử dụng các lệnh if…elif để thực hiện vai trò như của switch hoặc case như trong ví dụ trên.

* + 1. Vòng lặp ‘while’

Vòng lặp while trong Python thực thi lặp đi lặp lại các lệnh hoặc phần thân của vòng lặp miễn là điều kiện đã cho là true. Khi điều kiện là false, thì điều khiển sẽ thoát ra khỏi vòng lặp. Dưới đây là cú pháp của vòng lặp while trong Python:

While bieu\_thuc:

Cac\_lenh

Ở đây, cac\_lenh có thể là một lệnh đơn hoặc một khối lệnh. Bieu\_thuc có thể là bất kỳ biểu thức nào.

* + 1. Vòng lặp ‘for’

Vòng lặp for được sử dụng để lặp một biến qua một dãy (List hoặc String) theo thứ tự mà chúng xuất hiện. Sau đây là cú pháp của vòng lặp for:

for bien\_vong\_lap in day sequense:

cac\_lenh

Nếu một dãy day\_sequense gồm một danh sách các biểu thức, nó được ước lượng đầu tiên. Sau đó, item đầu tiên trong dãy được gán cho biến vòng lặp bien\_vong\_lap. Tiếp theo, các khối lệnh cac\_lenh được thực thi và khối lệnh này được thực thi tới khi dãy này đã được lặp xong.

Sử dụng lệnh else với vòng lặp for trong Python

* + 1. Vòng lặp lồng nhau

Ngôn ngữ lập trình Python cho phép bạn sử dụng một vòng lặp bên trong một vòng lặp khác hay còn gọi là **vòng lặp lòng nhau trong python**. Dưới đây là cú pháp và một số ví dụ.

Cú pháp lồng vòng lặp for trong Python

|  |
| --- |
| **for** bien\_vong\_lap\_1 **in** day\_seq\_1:  **for** bien\_vong\_lap\_2 **in** day\_seq\_2:  **cac\_lenh**      cac\_lenh |

Cú pháp lồng vòng lặp while trong Python

|  |
| --- |
| **while**(condition1):  **while**(condition2):  **//** Khối lệnh được lặp lại cho đến khi condition2 **=** False  **//** Khối lệnh được lặp lại cho đến khi condition1 **=** False |

* + 1. Các lệnh ‘break’, ‘continue’ và ‘pass’

Break: Break chấm dứt vòng lặp hiện tại và tiếp tục thực hiện tại câu lệnh tiếp theo, giống như câu lệnh break truyền thống trong C.

Việc sử dụng break là khi một số điều kiện bên ngoài được kích hoạt đòi hỏi phải vội vã rời khỏi một vòng lặp. Câu lệnh break có thể được sử dụng cả trong vòng lặp for và while.

Nếu đang sử dụng các vòng lặp lồng nhau, câu lệnh break sẽ ngừng việc thực hiện vòng lặp bên trong và bắt đầu thực thi dòng mã tiếp theo sau khối

Continue: Lệnh continue trả về điều khiển tới phần ban đầu của vòng lặp. Lệnh này bỏ qua lần lặp hiện tại và bắt buộc lần lặp tiếp theo của vòng lặp diễn ra. Lệnh continue có thể được sử dụng trong vòng lặp while hoặc vòng lặp for.

Pass: Lệnh pass, giống như tên của nó, được sử dụng khi một lệnh là cần thiết theo cú pháp nhưng không muốn bất cứ lệnh hoặc khối code nào được thực thi. Lệnh pass là một hoạt động null và không có gì xảy ra khi nó thực thi.

* 1. Hàm và Thủ Tục

Hàm là một khối mã có thể tái sử dụng, thực hiện một công việc cụ thể và **trả về một giá trị**. Trong Python, hàm được định nghĩa bằng từ khóa def, theo sau là tên hàm và các tham số trong dấu ngoặc tròn ( ). Sau khi thực hiện xong các tác vụ, hàm có thể sử dụng lệnh return để trả về một giá trị.

Cú pháp của hàm:

def ten\_ham(tham\_so\_1, tham\_so\_2):

# Khối mã

ket\_qua = tham\_so\_1 + tham\_so\_2

return ket\_qua

Thủ tục cũng là một khối mã có thể tái sử dụng, nhưng không **trả về giá trị**. Thủ tục thường được sử dụng khi chỉ cần thực hiện một số tác vụ mà không cần trả về kết quả. Trong Python, một thủ tục thường là một hàm không có return, hoặc có return nhưng không trả về giá trị nào cụ thể.

**Cú pháp của thủ tục:**

def ten\_thu\_tuc(tham\_so\_1, tham\_so\_2):

# Khối mã

print("Thực hiện một số tác vụ")

#### Thư Viện, Môđun và khung

* 1. Mô-đun

Mô-đun trong Python: Là tệp .py chứa mã nguồn bao gồm hàm, lớp, biến và có thể cả mã thực thi.

Mục đích: Tổ chức và tái sử dụng mã dễ dàng hơn, giúp lập trình viên tập trung vào từng phần nhỏ của ứng dụng.

Mô-đun Python phổ biến:

* random: Tạo số ngẫu nhiên và mẫu giả ngẫu nhiên.
* html: Phân tích và thao tác với tệp HTML.
* datetime: Xử lý ngày và giờ.
* re: Làm việc với biểu thức chính quy.

Lợi ích của mô-đun:

Cải thiện quy trình phát triển: Làm việc trên từng phần nhỏ giúp giảm lỗi và tăng tốc độ phát triển.

Tái sử dụng mã: Sử dụng lại mô-đun ở nhiều nơi, giảm trùng lặp và tăng hiệu quả.

Không gian tên riêng biệt: Tránh xung đột tên giữa các mô-đun và các phần khác trong ứng dụng.

* 1. Thư viện

Thư viện trong Python: Là tập hợp các mô-đun và gói hỗ trợ tái sử dụng mã, giúp phát triển ứng dụng nhanh hơn. Thư viện thường chứa nhiều gói theo chủ đề.

Phân biệt thư viện và gói:

Gói (package): Tập hợp các mô-đun.

Thư viện (library): Tập hợp các gói, thường chứa nhiều tính năng đa dạng.

Nguồn tải thư viện: PyPI (Python Package Index) là kho chứa thư viện Python phổ biến, hỗ trợ tải và cài đặt dễ dàng.

Thư viện Python phổ biến:

* Matplotlib: Tạo đồ thị và trực quan hóa dữ liệu.
* PyTorch: Thư viện học sâu, hỗ trợ lập trình tensor và xây dựng mô hình AI.
* pygame: Phát triển trò chơi, hỗ trợ âm thanh, hình ảnh, và sự kiện.
* Beautiful Soup: Phân tích, trích xuất dữ liệu từ HTML/XML (web scraping).
* Requests: Xử lý yêu cầu HTTP, làm việc với API dễ dàng.
* missingno: Trực quan hóa và xử lý dữ liệu bị thiếu, hỗ trợ dọn dẹp dữ liệu hiệu quả.

#### Ưu và Nhược Điểm

Ưu điểm của Python

* Dễ đọc và dễ học: Python có cú pháp đơn giản, dễ hiểu, giúp lập trình viên tập trung vào việc xây dựng logic hơn là xử lý cú pháp phức tạp.
* Giảm chi phí bảo trì: Python dễ bảo trì, từ đó giảm chi phí phát triển dài hạn do mã dễ đọc và dễ quản lý.
* An toàn trước lỗi phân đoạn: Python giảm thiểu khả năng xảy ra lỗi phân đoạn (segmentation faults), giúp ứng dụng ổn định hơn.
* Khả năng ứng dụng rộng rãi: Python được sử dụng rộng rãi trong khoa học, kỹ thuật, và toán học, phù hợp cho nhiều lĩnh vực ứng dụng.
* Quản lý bộ nhớ hiệu quả: Python có hệ thống quản lý bộ nhớ tích hợp với heap riêng và cơ chế thu gom rác (garbage collection), giúp sử dụng bộ nhớ hiệu quả hơn.
* Đơn giản và nhanh chóng: Python được xem là "ngôn ngữ sẵn sàng chạy" với cú pháp đơn giản, dễ kiểm tra và triển khai nhanh chóng.
* Hỗ trợ mã hóa không đồng bộ: Python hỗ trợ mã không đồng bộ với vòng lặp sự kiện, giúp cải thiện hiệu suất trong xử lý đồng thời.
* Tích hợp dễ dàng với các ngôn ngữ khác: Thông qua các thư viện như Cython và Jython, Python có thể tích hợp với C, C++, và Java, hỗ trợ phát triển đa nền tảng.
* Tích hợp ứng dụng doanh nghiệp: Python hỗ trợ Tích hợp ứng dụng doanh nghiệp (EAI) với tính năng kiểm soát quy trình, tương thích với nhiều giao thức dữ liệu internet và có thể dùng như ngôn ngữ kịch bản.

Nhược điểm của Python

* Tốc độ thực thi chậm:Python là ngôn ngữ thông dịch, chậm hơn các ngôn ngữ biên dịch như C++ và Java.
* Tiêu thụ bộ nhớ lớn: Python yêu cầu nhiều bộ nhớ, không phù hợp với các ứng dụng cần tối ưu bộ nhớ.
* Không phù hợp cho phát triển di động và trò chơi: Python chủ yếu được dùng cho máy tính để bàn và web, không lý tưởng cho ứng dụng di động và trò chơi do tiêu tốn nhiều bộ nhớ và tốc độ xử lý chậm.
* Hạn chế về trải nghiệm lập trình: Các lập trình viên có thể gặp khó khăn khi chuyển sang ngôn ngữ khác sau khi quen với sự đơn giản của Python.
* Khó phát hiện lỗi trong quá trình phát triển: Python không hỗ trợ phát hiện lỗi trong quá trình biên dịch, dễ gây khó khăn trong việc gỡ lỗi.
* Hạn chế truy cập cơ sở dữ liệu: Lớp truy cập cơ sở dữ liệu của Python chưa phát triển bằng các công nghệ như JDBC và ODBC, gây hạn chế khi làm việc với cơ sở dữ liệu.
* Giới hạn trong thiết kế: Python có một số hạn chế trong thiết kế, làm giảm tính linh hoạt so với các ngôn ngữ lập trình khác.
* Khó khăn trong kiểm tra mã: Các lỗi chỉ được phát hiện trong quá trình chạy, gây khó khăn trong việc kiểm tra và phát hiện lỗi sớm.

#### Các Ứng Dụng Thực Tế

* Phát triển Web: Sử dụng Django, Flask để xây dựng ứng dụng web.
* Phân tích Dữ liệu: Xử lý và trực quan hóa dữ liệu với Pandas, NumPy, Matplotlib.
* AI và Học máy: Xây dựng mô hình AI với TensorFlow, PyTorch.
* Ứng dụng Máy tính để bàn: Tạo giao diện người dùng với Tkinter, PyQt.
* Tự động hóa: Tự động hóa các tác vụ và quản lý hệ thống.
* Phát triển Trò chơi: Tạo trò chơi đơn giản với Pygame.
* IoT: Chạy trên các thiết bị như Raspberry Pi.
* Xử lý Ngôn ngữ Tự nhiên: Phân tích văn bản với NLTK, spaCy.
* Web Scraping: Thu thập dữ liệu web với Beautiful Soup, Scrapy.
* Tài chính: Phân tích tài chính và dự đoán.
* Y sinh học: Phân tích dữ liệu sinh học với Biopython.
* Tự động hóa và Scripting: Python rất phù hợp cho tự động hóa các tác vụ, từ việc tự động hóa trên máy tính cá nhân (như quản lý tệp và tạo báo cáo) đến các quy trình công nghiệp phức tạp.
* Phát triển Ứng dụng Khoa học và Tính Toán: Python được sử dụng trong các ứng dụng khoa học, kỹ thuật và tính toán số nhờ khả năng tích hợp với các thư viện hiệu năng cao như NumPy và SciPy.
* Lập trình Hệ Thống và DevOpsz: Python được sử dụng để viết các công cụ DevOps, tập lệnh triển khai và các ứng dụng quản lý hệ thống. Ansible và SaltStack là ví dụ của các công cụ tự động hóa sử dụng Python.

# CHƯƠNG 2: PHÂN TÍCH & THIẾT KẾ HỆ THỐNG BẰNG DJANGO TRONG PYTHON

### **Phân tích yêu cầu hệ thống**

#### Yêu cầu chức năng

Hệ thống website mạng xã hội sử dụng Python Django cần đáp ứng các yêu cầu chức năng sau:

-**Đăng ký và Đăng nhập:**

+ Đăng ký tài khoản**:** Người dùng có thể tạo tài khoản bằng email, số điện thoại, hoặc đăng nhập qua tài khoản mạng xã hội khác (Google, Facebook)..

+ Xác thực và bảo mật:Tích hợp xác thực hai yếu tố (2FA),gửi email xác nhận khi đăng ký tài khoản.

+ Phân quyền: User normal và Admin.

**- Quản lý hồ sơ người dùng (User Profile):**

**+** Cho phép người dùng chỉnh sửa thông tin cá nhân như ảnh đại diện, tên, tiểu sử, v.v.

**+** Cài đặt quyền riêng tư (công khai, chỉ bạn bè, riêng tư).

+ Hiển thị danh sách bạn bè hoặc người theo dõi.

**- Quản trị hệ thống:**

**+** Giao diện quản trị cho phép quản lý tài khoản người dùng, bài viết, nhóm, và báo cáo

**+** Cung cấp các công cụ thống kê để phân tích dữ liệu người dùng và tương tác.

+Quản lý Admin quản lý người dùng và các bài viết người dùng đăng.

**-** **Tích hợp quảng cáo và thương mại:**

**+** Hiển thị quảng cáo dựa trên hành vi và sở thích người dùng.

**+** Tích hợp hệ thống thanh toán nếu hỗ trợ mua sắm hoặc giao dịch.

- **Báo cáo và kiểm duyệt nội dung:**

**+** Cung cấp chức năng báo cáo nội dung vi phạm hoặc tài khoản không phù hợp.

+ Quản trị viên có quyền xử lý báo cáo và gỡ bỏ nội dung nếu cần thiết.

#### Yêu cầu phi chức năng

**- Bảo mật:**

**+Bảo vệ dữ liệu cá nhân**: Mã hóa mật khẩu bằng thuật toán mạnh như bcrypt,sử dụng HTTPS để mã hóa dữ liệu truyền tải.

+ Đảm bảo tính bảo mật không còn các nỗi lo về các lỗi bảo mật thông thường. Django cũng cung cấp cả phương pháp để lưu mật khẩu an toàn

**+ Sử dụng HTTPS:** Tất cả các trang web phải sử dụng HTTPS để bảo vệ thông tin người dùng.

**+ Phòng chống tấn công**: Tích hợp các biện pháp chống SQL Injection, Cross-Site Scripting (XSS), và Cross-Site Request Forgery (CSRF).

+**Quản lý quyền truy cập**:Chỉ những người có quyền mới được truy cập vào các khu vực quản trị hoặc dữ liệu quan trọng.

**- Tương thích (Compatibility):**

**+** Hệ thống phải hoạt động tốt trên nhiều trình duyệt phổ biến (Chrome, Firefox, Safari, Edge).

**+** Tương thích với nhiều thiết bị (máy tính, điện thoại, máy tính bảng).

**-Tuân thủ pháp luật và quy định (Compliance):**

**+** Tuân thủ các quy định bảo vệ dữ liệu cá nhân như **GDPR**, **CCPA**, và các quy định địa phương khác.

**+** Đảm bảo quyền riêng tư và cung cấp chính sách bảo mật minh bạch.

**-Tính đo lường và phân tích (Analytics):**

**+** Hỗ trợ thu thập và phân tích dữ liệu người dùng để đưa ra các gợi ý, tối ưu hóa trải nghiệm.

+ Báo cáo chi tiết về số lượng người dùng, mức độ tương tác và xu hướng sử dụng.

### **Thiết kế kiến trúc hệ thống**

#### Kiến trúc tổng thể

Hệ thống được chia thành nhiều lớp để đảm bảo tính mô-đun và quản lý:

**Lớp giao diện người dùng (Presentation Layer)**

- **Thành phần:**

+ Giao diện web (HTML, CSS, JavaScript).

+ Các công cụ front-end như React.js, Vue.js hoặc Bootstrap để tạo giao diện thân thiện.

-**Chức năng:**

**+** Hiển thị nội dung cho người dùng.

**+** Tương tác với API để gửi và nhận dữ liệu từ lớp logic ứng dụng.

**Lớp logic ứng dụng (Application Logic Layer):**

**-Thành phần**:

+Django Framework:

+Xử lý các yêu cầu từ giao diện người dùng.

+Tích hợp các ứng dụng như quản lý người dùng, bài viết, tin nhắn, thông báo.

+Django REST Framework (DRF): Cung cấp API RESTful để giao tiếp với các ứng dụng front-end hoặc mobile app.

**- Chức năng:**

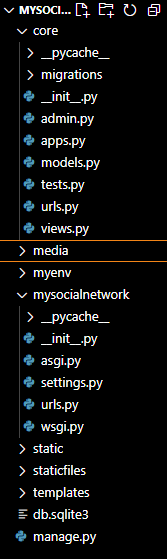
+ Xử lý logic nghiệp vụ như xác thực người dùng, kết nối bạn bè, hoặc quản lý nội dung.

+ Quản lý các tác vụ nền bằng Celery (ví dụ: gửi email, thông báo).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| | Thành phần |  |  | | --- | --- | --- | | Công nghệ đề xuất |
| Framework Back-end | Python Django, DRF |
| Front-end | HTML/CSS, React.js/Vue.js |
| Cơ sở dữ liệu | PostgreSQL, Redis |
| Lưu trữ tệp | Amazon S3, Google Cloud |
| Tác vụ nền | Celery, RabbitMQ |
| WebSocket (nếu cần real-time) | Django Channels |
| Triển khai | Docker, Kubernetes |

Cấu trúc dự án :

-Thành phần



-Chức năng:

+ Khởi động máy chủ phát triển:python manage.py runserver

+ Tạo và áp dụng các di chuyển: python manage.py makemigrationsvà `mmigrate

+ Tạo người dùng quản trị:python manage.py createsuperuser

+ Tạo các tùy chỉnh lệnh.

### **Sơ đồ kiến trúc hệ thống**

Hệ thống gồm các thành phần chính:

1. **Frontend (Giao diện người dùng)**: Giao diện nơi người dùng tương tác (web hoặc ứng dụng di động).**Server** → Xử lý yêu cầu HTTP / HTTPS với Django.
2. **Backend (Hệ thống phía server)**: Nơi xử lý logic và kết nối dữ liệu.
3. **Database (Cơ sở dữ liệu)**: Lưu trữ thông tin người dùng, bài viết, hình ảnh.
4. **Media Storage (Lưu trữ)**: Nơi lưu hình ảnh, video.

### **Sơ đồ luồng dữ liệu**

Sơ đồ luồng dữ liệu (DFD) cho website trang mạng xã hội sẽ mô tả cách dữ liệu di chuyển giữa các thành phần chính trong hệ thống như người dùng, frontend, backend, cơ sở dữ liệu và lưu trữ được thể hiện như sau :

**- Người dùng** tương tác với **Frontend**: Đăng nhập, đăng bài, bình luận, v.v.

- **Frontend** gửi yêu cầu đến **Backend** để xử lý các hành động của người dùng.

**- Backend** tương tác với **Database** để lưu trữ hoặc truy xuất dữ liệu (như thông tin người dùng, bài viết, bình luận).

**- Backend** tương tác với **Media Storage** để lưu hoặc lấy hình ảnh/video.

- **Frontend** hiển thị kết quả từ **Backend** (như thông báo thành công, bài viết mới, hình ảnh/video).

- **Người dùng** nhận kết quả và tiếp tục tương tác với hệ thống.

### **Thiết kế giao diện người dùng (UI)**

#### Nguyên tắc thiết kế

- **Thân thiện với người dùng**: Giao diện nên rõ ràng và dễ hiểu với người dùng. Tránh sự phức tạp không cần thiết.

**Cách thực hiện**: Sử dụng không gian trống hợp lý (white space), các yếu tố phải nổi bật, dễ nhận diện và tránh sử dụng quá nhiều màu sắc hoặc kiểu chữ.

- **Tính nhất quán**: Giao diện cần phải nhất quán trên toàn bộ trang web hoặc ứng dụng để người dùng dễ dàng hiểu và sử dụng.

**Cách thực hiện**: Sử dụng các thành phần giao diện đồng nhất, như bảng màu, kiểu chữ, nút, biểu tượng, và layout.

-**Khả năng đáp ứng**:Giao diện cần phải tương thích với mọi kích thước màn hình (desktop, tablet, smartphone).

**Cách thực hiện**: Sử dụng thiết kế web đáp ứng (responsive design) và kiểm tra trên các loại màn hình khác nhau.

-**Trải nghiệm người dùng**:Thiết kế UI không chỉ đẹp mắt mà còn phải mang lại trải nghiệm người dùng tuyệt vời

**Cách thực hiện:** Đảm bảo quy trình đăng nhập, đăng bài, tìm kiếm và các chức năng khác dễ dàng, không gây khó khăn cho người dùng.

**-Tương tác trực quan**:Cung cấp các phản hồi trực quan khi người dùng thực hiện các hành động (nhấn nút, di chuột, v.v.)

**Cách thực hiện**: Sử dụng các hiệu ứng hover, màu sắc thay đổi khi chọn lựa, nút bấm rõ ràng, thông báo cho người dùng khi thao tác hoàn tất.

-**Tập trung vào người dùng**:Thiết kế giao diện phải phù hợp với nhu cầu và mong muốn của người dùng.

**Cách thực hiện**: Nghiên cứu người dùng mục tiêu và dựa trên dữ liệu đó để tối ưu hóa giao diện (ví dụ: thiết kế cho người dùng trẻ tuổi hay người dùng lớn tuổi sẽ khác nhau).

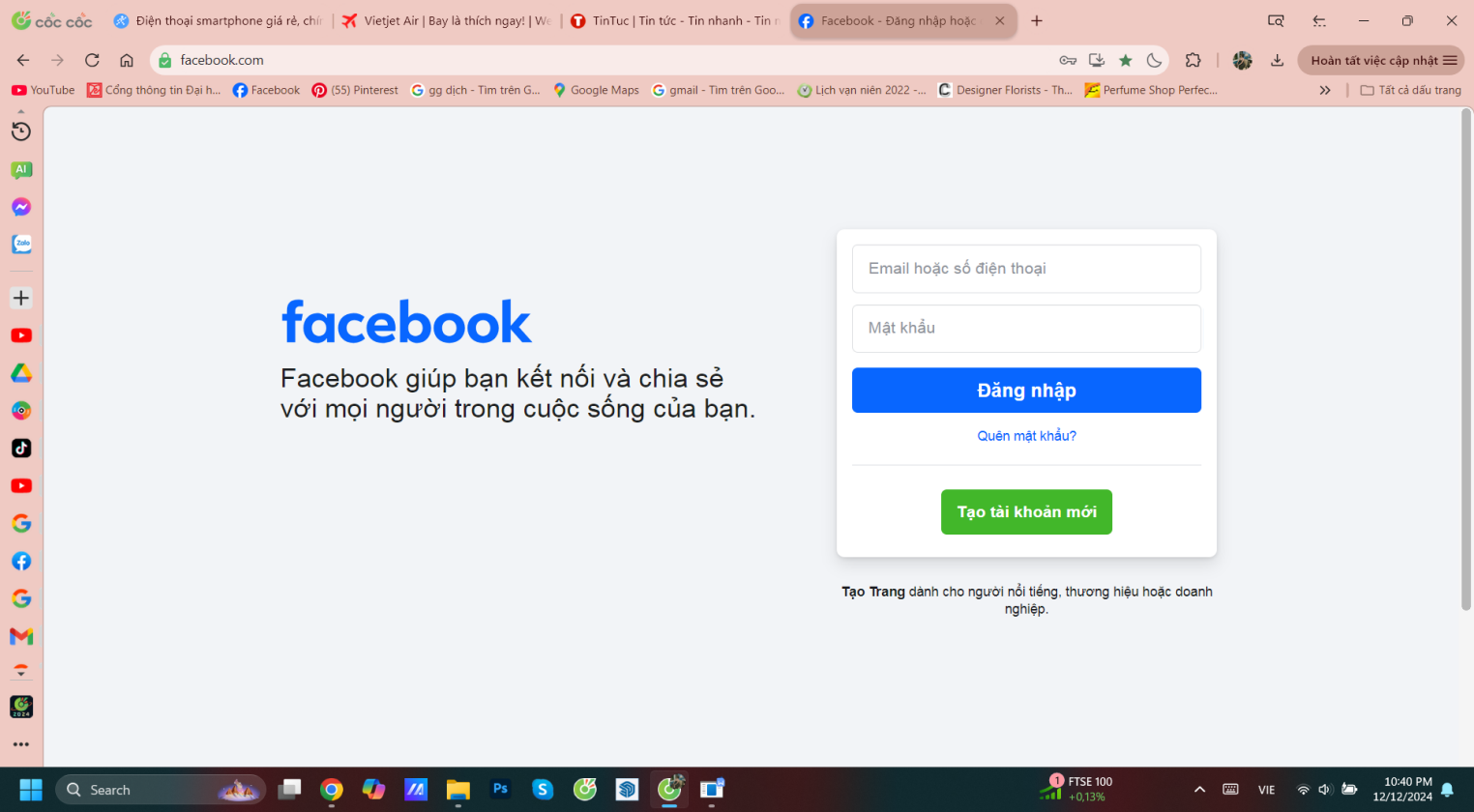
-**Phản hồi và Hướng dẫn**:Cung cấp thông tin phản hồi khi người dùng thực hiện hành động và hướng dẫn khi cần thiết.

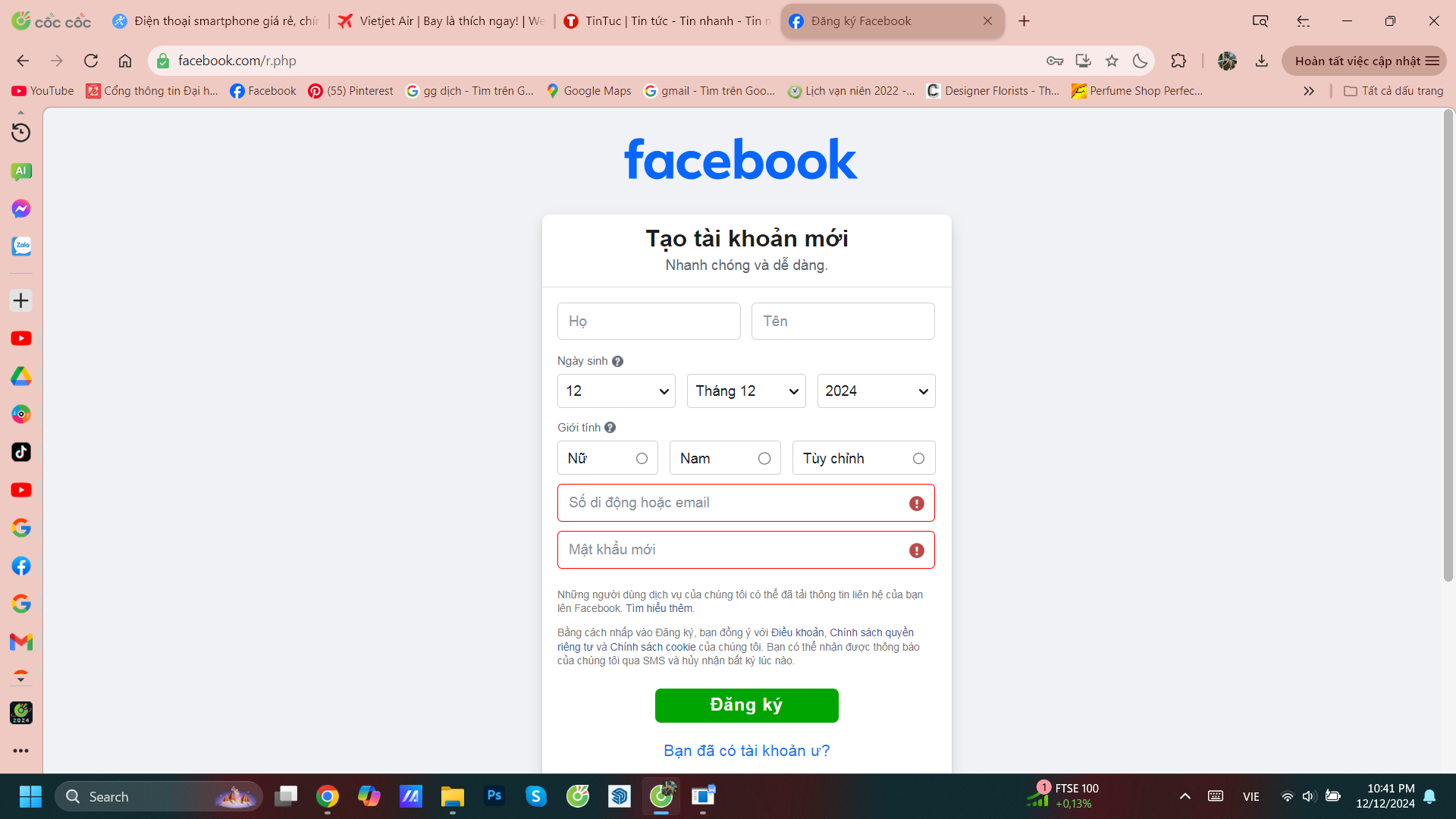
**+Cách thực hiện**: Cung cấp thông báo rõ ràng về lỗi (nếu có), trạng thái của yêu cầu, và sử dụng các thông báo để hướng dẫn người dùng.

#### Mẫu giao diện

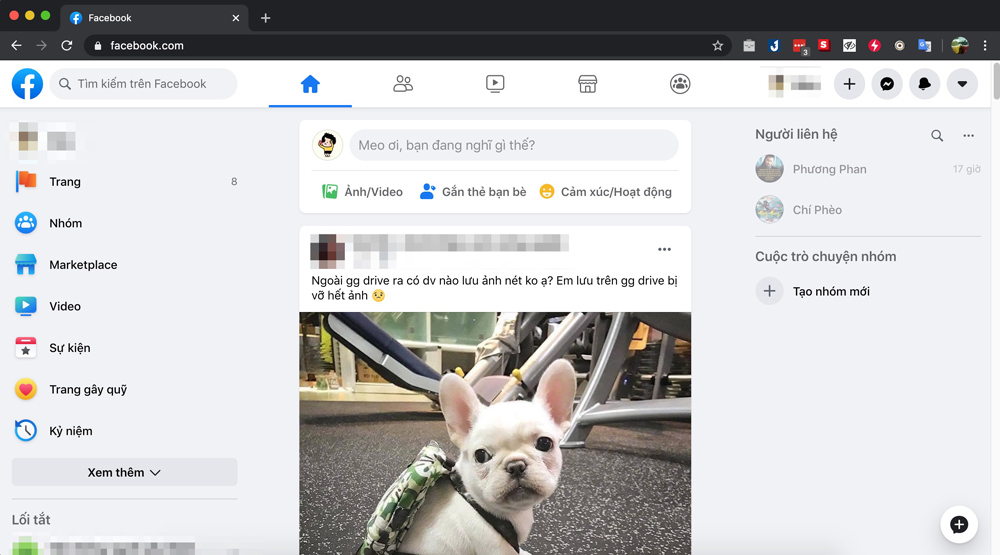
Dưới đây là một số mẫu giao diện cho website trang mạng xã hội:

**- Trang đăng nhập, đăng ký:** Cho phép người dùng đăng nhập hoặc tạo tài khoản mới, với các trường thông tin cần thiết và liên kết đến chức năng khôi phục mật khẩu.

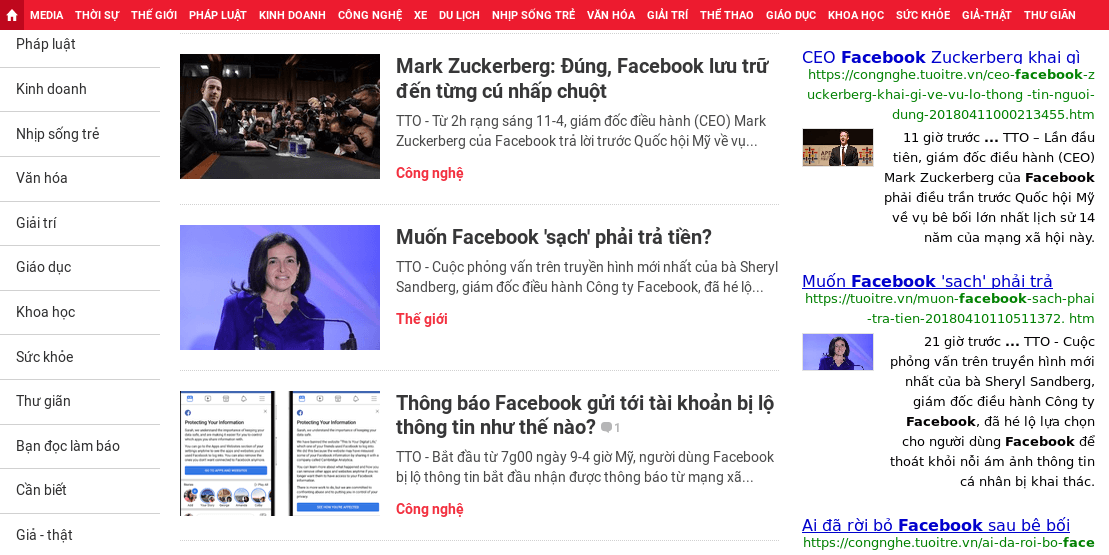




**- Trang chủ:** Hiển thịnhững thông tin liên quan khi bạn tìm kiếm , các chương trình khuyến mãi, và các danh mục sản phẩm,tin tức ,kênh giải trí ,học tập ,.. chính để người dùng dễ dàng tìm kiếm.



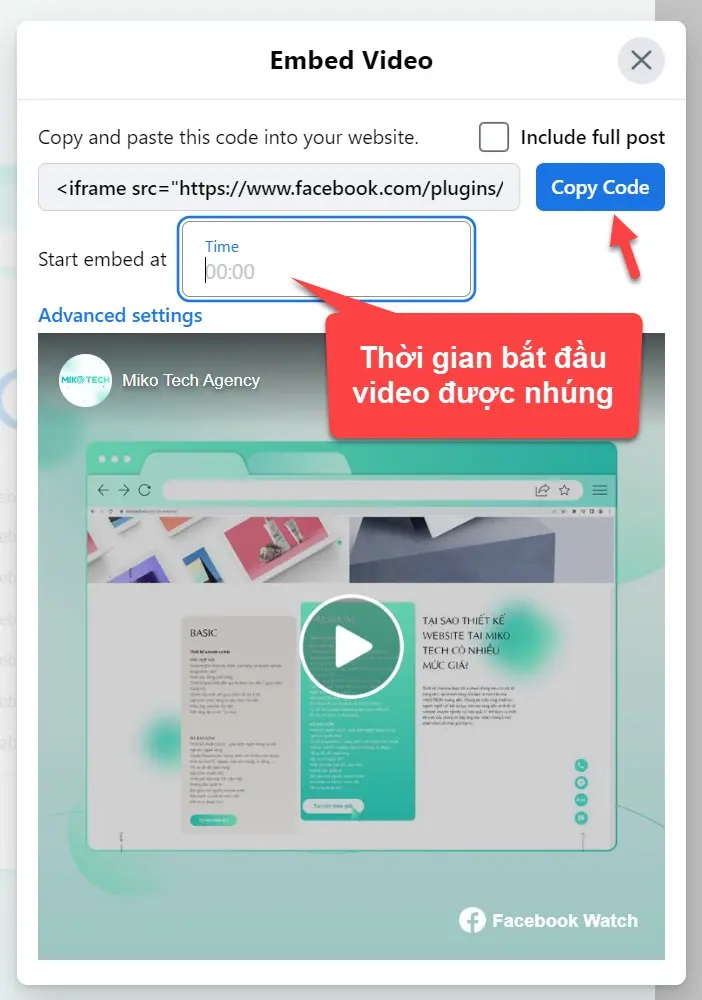
**- Trang tin tức** : Cung cấp các tin tức , bao gồm hình ảnh, video,...



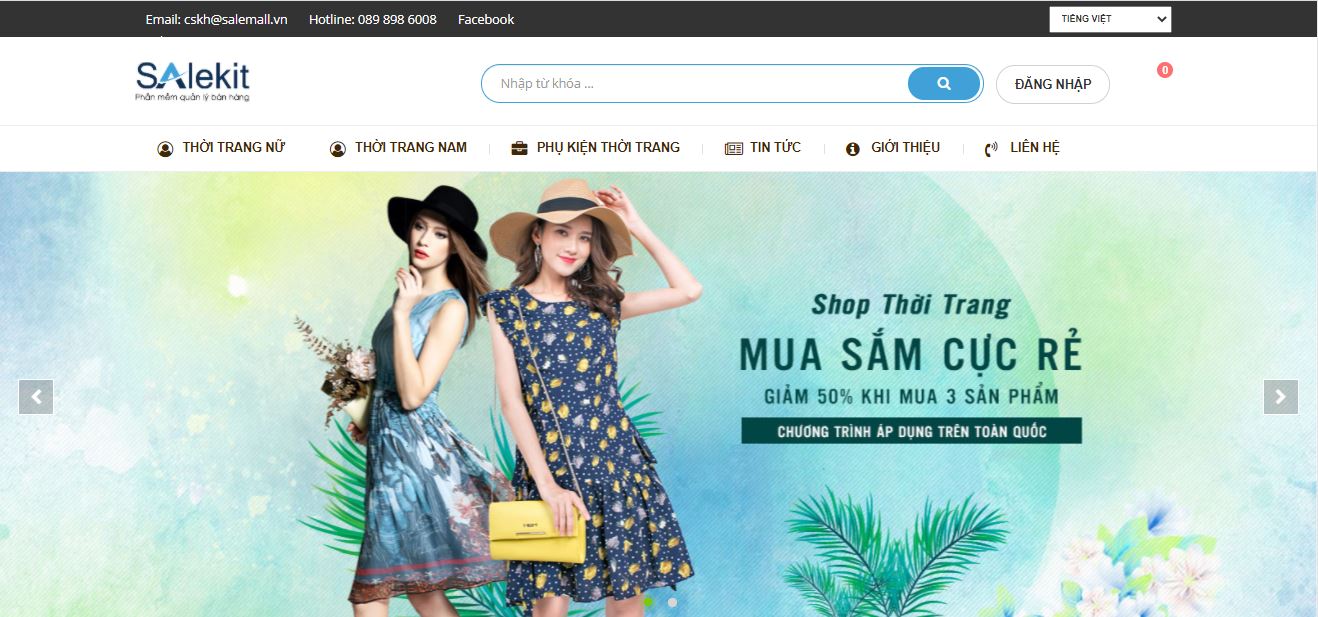
**- Trang chi tiết**: Cung cấp thông tin chi tiết về sản phẩm ol,tin tức giải trí , bao gồm mô tả, hình ảnh, giá cả.



**- Trang video:** Phân chia ra các video ,tin tức ,sản phẩm mà bạn tìm kiếm một cách hiệu quả và nhanh chóng nhất có thể .



**- Trang tìm kiếm:** Cung cấp chức năng tìm kiếm bạn bè ,kênh giải trí ,tin tức ,...



### **Thiết kế cơ sở dữ liệu**

#### Mô hình dữ liệu

Mô hình dữ liệu là một phần quan trọng trong thiết kế cơ sở dữ liệu cho một hệ thống mạng xã hội. Mô hình này sẽ xác định các thực thể (entities) trong hệ thống và các mối quan hệ giữa chúng:

**- User (Người dùng):** Chứa thông tin về người dùng như tên, email, mật khẩu, ảnh đại diện, ngày tham gia, v.v

**- Post (Bài viết):** Chứa nội dung bài viết của người dùng, có thể bao gồm văn bản, hình ảnh, video

**-Comment (Bình luận):** Chứa các bình luận mà người dùng đăng dưới các bài viết.

-**Like (Lượt thích):**Chứa thông tin về việc người dùng thích bài viết.

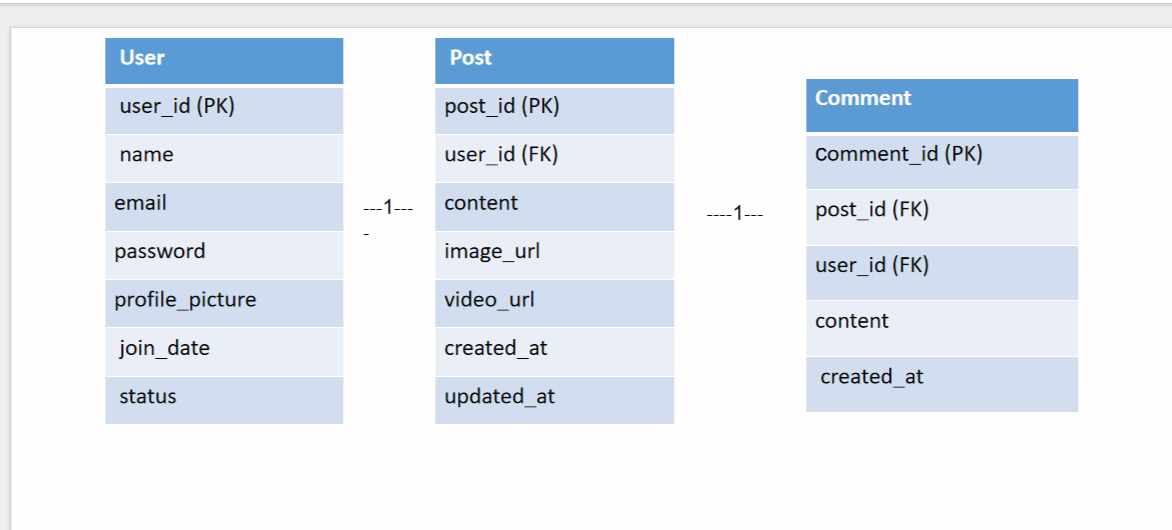
#### Sơ đồ thiết kế cơ sở dữ liệu

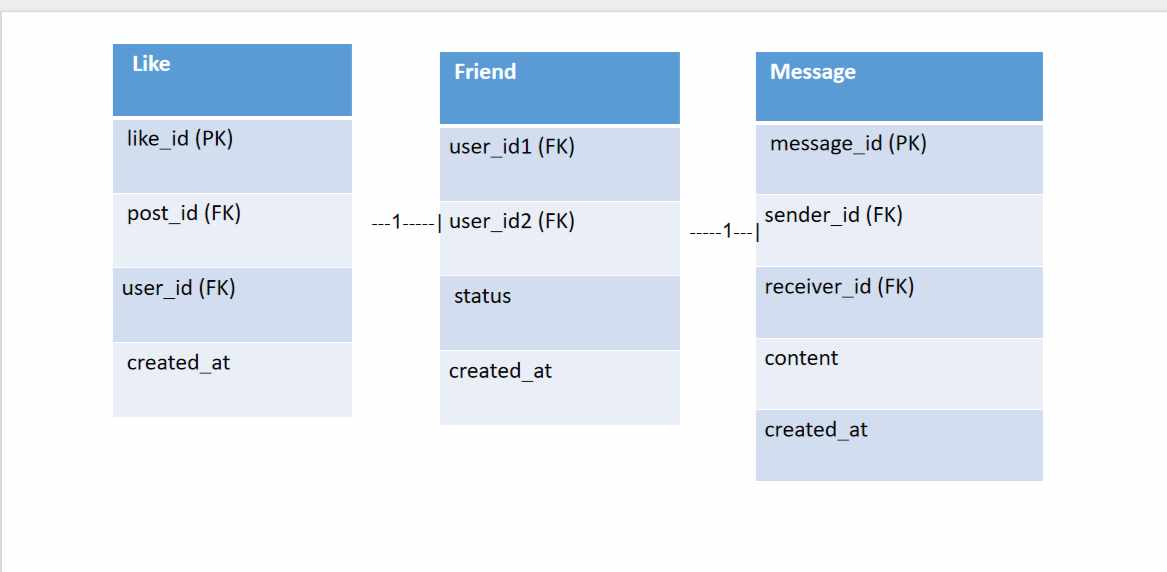
Sơ đồ thiết kế cơ sở dữ liệu sẽ mô tả các mối quan hệ giữa các bảng, giúp dễ dàng hình dung được các dữ liệu trong hệ thống mạng xã hội Ví dụ:

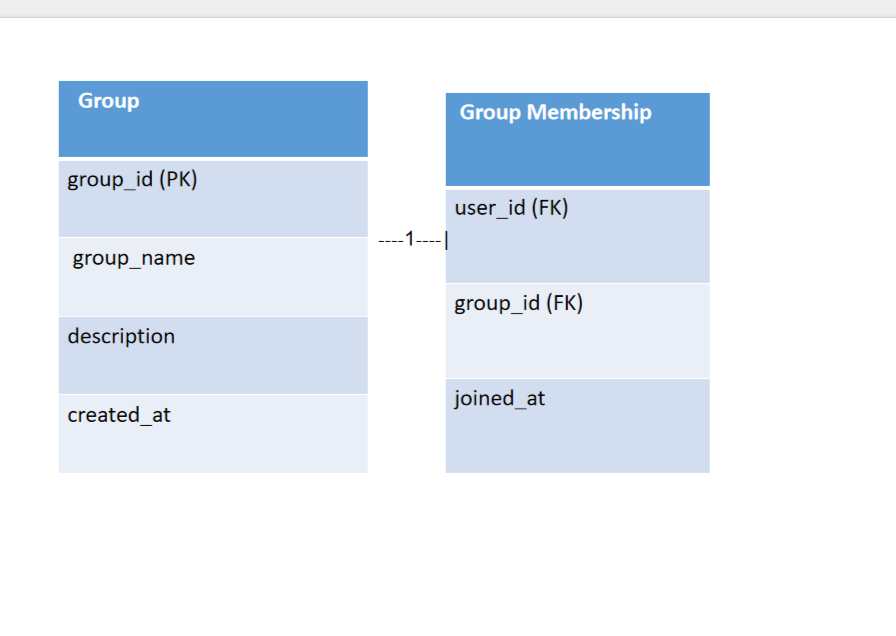
- Mối quan hệ **User (Người dùng)**- **Post(Bài viết)-** Comment (Bình luận)-**Like (Lượt thích)-**Friend (Bạn bè)

**Mối quan hệ giữa các bảng:**

* **User** có thể tạo nhiều **Post**, **Comment**, và **Like**.
* **Post** có thể có nhiều **Comment** và **Like**.
* **User** có thể có nhiều **Friend** (mối quan hệ bạn bè) và mỗi mối quan hệ có thể có trạng thái (chờ, kết bạn, v.v.).
* **User** có thể tham gia nhiều **Group**.
* **User** có thể gửi và nhận **Message** từ các người dùng khác.







### **Kết luận chương 2**

Chương này đã phân tích và thiết kế hệ thống website mạng xã hội môt cách chi tiết từ yêu cầu chức năng đến phi chức năng cho đến kiến trúc hệ thống ,thiết kế gio diện người dùng và cơ sở dữ liệu .Những yếu tố này đã đóng góp vai trò quan trọng trong việc xây dựng hệ thống website trang mạng xã hội một cách hiệu quả và đáp ứng được nhu cầu và mong muốn của người dùng ,tối ưu sự trải nghiệm xem tin tức,kinh doanh ol,học tập ,giải trí ,...Việc tạo nên một website mạng xã hội một cách cụ thể và hiểu rõ được các bước tiến hành giúp cho việc phát triển trang website một cách hiệu quả, nhanh chóng hơn ,hiệu quả hơn.

# CHƯƠNG 3: TRIỂN KHAI VÀ KIỂM THỬ HỆ THỐNG

### **Công cụ sử dụng**

- Ngôn ngữ lập trình:

+ Python: Sử dụng Django cho backend để xử lý logic nghiệp vụ.

+ HTML/CSS/JavaScript: Sử dụng các công nghệ này cho frontend, giúp tạo giao diện người dùng thân thiện.

- Cơ sở dữ liệu:

+ SQLite3: Dùng cho môi trường phát triển nhanh chóng.

+ MySQL: Sử dụng cho môi trường sản xuất để đảm bảo hiệu suất và khả năng mở rộng.

- Thư viện:

+ Django Rest Framework: Dùng để xây dựng API cho frontend.

+ Bootstrap: Sử dụng để thiết kế giao diện người dùng responsive và dễ sử dụng.

+ Pillow: Xử lý ảnh trong Python.

+ bcrypt: Mã hóa mật khẩu.

+ django-crispy-forms: Cải thiện giao diện form trong Django.

+ Turial: Công cụ theo dõi hoặc tự động hóa các tác vụ

+ tzdata: Hỗ trợ múi giờ chính xác.

Cách tạo 1 bài làm:

1. Tạo môi trường làm việc

Bước 1: Cài đặt Python

python --version

Bước 2: Tạo môi trường ảo (Virtual Environment) Tạo môi trường ảo để cô lập các gói thư viện:

python -m venv venv

Bước 3: Kích hoạt môi trường ảo:

venv\Scripts\activate

2. Cài đặt Django

Bước 1: Cài đặt Django

- Cài đặt Django bằng pip: pip install django

- Kiểm tra phiên bản Django: python -m django --version

Bước 2: Tạo dự án Django

- Tạo dự án mới: [tên dự án: đặt tên django\_lab1]

django-admin startproject django\_lab1

- Tạo 1 ứng dụng [module]:

python manage.py startapp user\_management

- Di chuyển vào thư mục dự án:

cd django\_lab1

Bước 3: Chạy thử dự án

- Khởi động server để kiểm tra:

python manage.py runserver

- Truy cập vào trình duyệt tại: <http://127.0.0.1:8000>.

### **Triển khai hệ thống**

#### Kịch bản thứ 1: Mô-đun Đăng Ký Người Dùng

* Tính năng cần có

Người dùng nhập thông tin để đăng ký tài khoản (username, email, mật khẩu, xác nhận mật khẩu).

Kiểm tra thông tin hợp lệ (email đúng định dạng, mật khẩu đủ mạnh).

Mã hóa mật khẩu trước khi lưu vào cơ sở dữ liệu.

Xác minh qua email (nếu cần).

* Mô hình cơ sở dữ liệu

Model.py:

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

bio = models.TextField(blank=True)

def \_\_str\_\_(self):

        return self.user.username

* Xử lý logic đăng ký (Views)

Views.py:

def signup(request):

if request.method == 'POST':

username = request.POST.get('username')

email = request.POST.get('email')

password = request.POST.get('password')

password2 = request.POST.get('password2')

if password == password2:

if User.objects.filter(email=email).exists():

messages.error(request, 'Email đã được sử dụng')

return redirect('signup')

elif User.objects.filter(username=username).exists():

messages.error(request, 'Tên người dùng đã tồn tại')

return redirect('signup')

else:

user = User.objects.create\_user(username=username, email=email, password=password)

user.save()

user\_login = auth.authenticate(username=username, password=password)

auth.login(request, user\_login)

new\_profile = Profile.objects.create(user=user) # Không cần id\_user

new\_profile.save()

return redirect('settings')

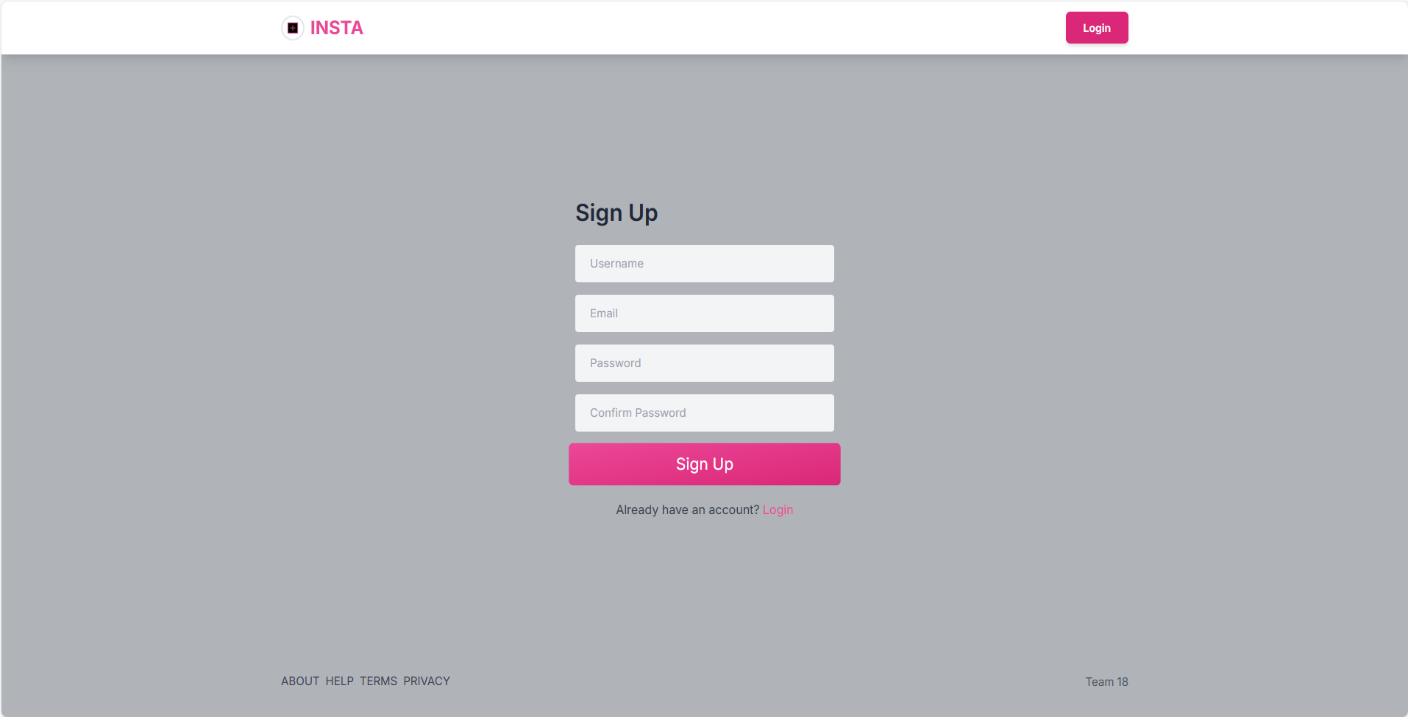
else:

messages.error(request, 'Mật khẩu không khớp')

return redirect('signup')

return render(request, 'signup.html')

* Giao diện



#### Kịch bản 2: Mô-đun Đăng Nhập Người Dùng (User Login)

* Tính năng cần có

Người dùng nhập thông tin đăng nhập (tên người dùng hoặc email và mật khẩu).

Xác thực thông tin với cơ sở dữ liệu.

Báo cáo lỗi nếu thông tin không chính xác.

* Mô hình cơ sở dữ liệu

Model.py:

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

bio = models.TextField(blank=True)

def \_\_str\_\_(self):

        return self.user.username

* Xử lý logic đăng ký (Views)

Views.py:

def signin(request):

if request.method == 'POST':

username = request.POST.get('username')

password = request.POST.get('password')

user = auth.authenticate(username=username, password=password)

if user is not None:

auth.login(request, user)

return redirect('/') # Điều hướng đến trang chủ

else:

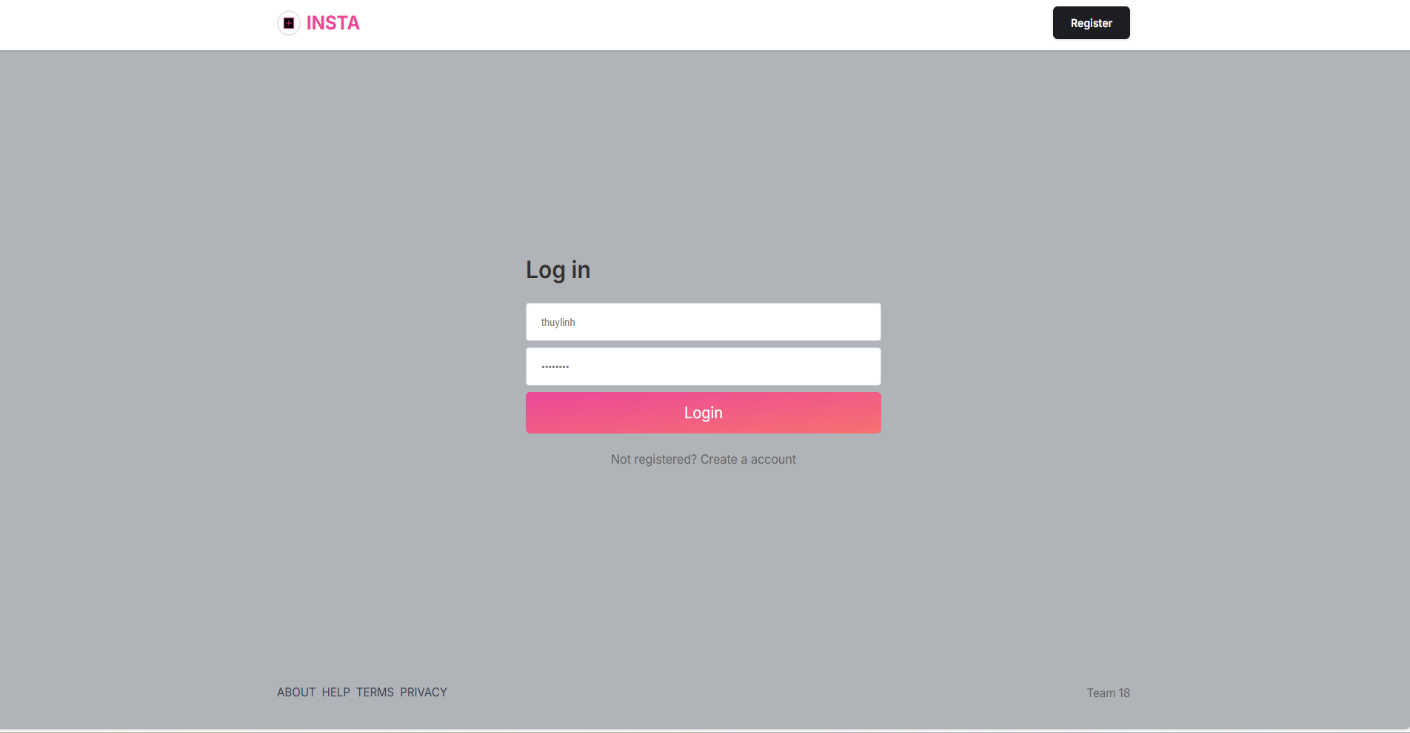
messages.error(request, 'Thông tin đăng nhập không hợp lệ')

return redirect('signin') # Quay lại trang đăng nhập

# Hiển thị trang đăng nhập nếu phương thức GET

return render(request, 'signin.html')

* Giao diện



#### Kịch bản 3: Cấu hình account settings

* Tính năng cần có

Profile Image: Tải hình đại diện.

Bio: Phần thông tin mô tả người dùng.

Location: Vị trí của người dùng.

* Mô hình cơ sở dữ liệu

Model.py:

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

bio = models.TextField(blank=True)

profileimg = models.ImageField(upload\_to='profile\_images', default='andanh.jpg')

location = models.CharField(max\_length=100, blank=True)

def \_\_str\_\_(self):

return self.user.username

* Xử lý logic đăng ký (Views)

Views.py:

def settings(request):

# Lấy thông tin profile của người dùng

user\_profile = Profile.objects.get(user=request.user)

if request.method == 'POST':

# Nếu không có ảnh mới thì giữ nguyên ảnh cũ

if request.FILES.get('image') is None:

image = user\_profile.profileimg

else:

# Nếu có ảnh mới, lấy ảnh từ request

image = request.FILES.get('image')

# Lấy bio và location từ form

bio = request.POST['bio']

location = request.POST['location']

# Cập nhật thông tin profile

user\_profile.profileimg = image

user\_profile.bio = bio

user\_profile.location = location

# Lưu lại thay đổi

user\_profile.save()

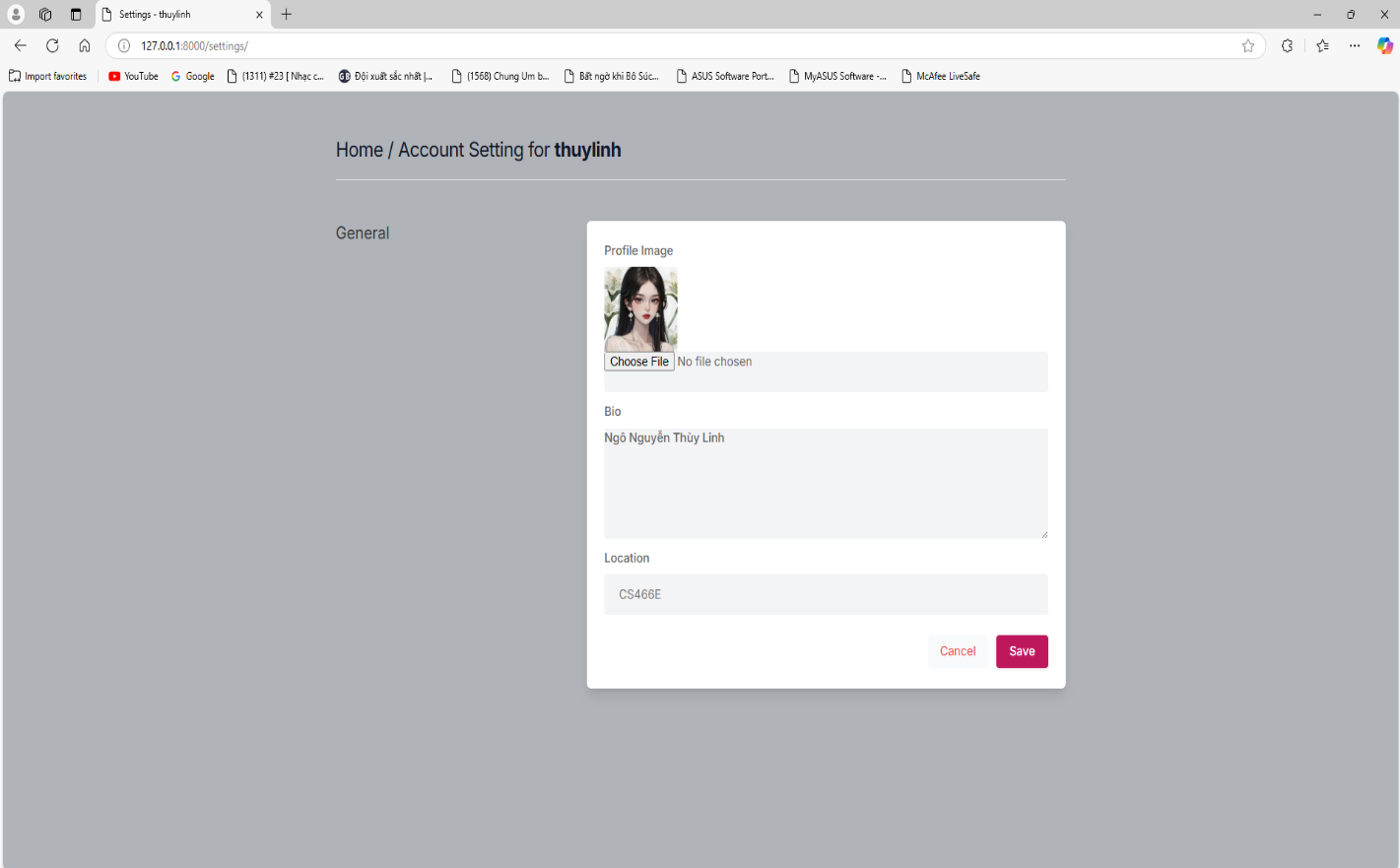
# Sau khi lưu, bạn có thể render lại trang settings

return redirect('settings')

# Phương thức GET (khi chưa submit form), hiển thị thông tin profile hiện tại

return render(request, 'setting.html', {'user\_profile': user\_profile})

* Giao diện



#### Kịch bản 4: Cấu hình trang chủ

* Tính năng cần có

Trong Django, trang chủ (home page) thường là nơi hiển thị các thông tin nổi bật, như danh sách bài viết, bài viết nổi bật, thông tin giới thiệu, like post, tải post, xóa post và thanh tìm kiếm

* Mô hình cơ sở dữ liệu

Model.py:

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE) # Liên kết một-một với User

bio = models.TextField(blank=True)

profileimg = models.ImageField(upload\_to='profile\_images', default='andanh.jpg')

location = models.CharField(max\_length=100, blank=True)

def \_\_str\_\_(self):

return self.user.username

class Post(models.Model):

id = models.UUIDField(primary\_key=True, default=uuid.uuid4, editable=False) # Sửa thành UUIDField

user = models.CharField(max\_length=100)

image = models.ImageField(upload\_to='post\_images')

caption = models.TextField()

created\_at = models.DateTimeField(default=datetime.now)

no\_of\_likes = models.IntegerField(default=0)

def \_\_str\_\_(self):

return self.user

class LikePost(models.Model):

post\_id = models.CharField(max\_length=500)

username = models.CharField(max\_length=100)

def \_\_str\_\_(self):

return self.username

* Xử lý logic đăng ký (Views)

Views.py:

@login\_required(login\_url='signin')

def index(request):

    user\_object = User.objects.get(username=request.user.username)  # Lấy đối tượng User

    user\_profile = Profile.objects.get(user=user\_object)  # Lấy Profile của user

    user\_following = FollowersCount.objects.filter(follower=request.user.username)

    user\_following\_list = []

    feed = []

    for users in user\_following:

        user\_following\_list.append(users.user)

    for usernames in user\_following\_list:

        feed\_lists = Post.objects.filter(user=usernames)

        feed.append(feed\_lists)

    feed\_list = list(chain(\*feed))

    post = Post.objects.all()

    return render(request, 'index.html', {'user\_profile': user\_profile, 'posts': post})  # Truyền user\_profile vào template

@login\_required(login\_url='signin')

def search(request):

    user\_object = User.objects.get(username=request.user.username)

    user\_profile = Profile.objects.get(user=user\_object)

    if request.method == 'POST':

        username = request.POST['username']

        username\_object = User.objects.filter(username\_\_icontains=username)

        username\_profile = []

        username\_profile\_list = []

        for users in username\_object:

            username\_profile.append(users.id)

        for ids in username\_profile:

            profile\_lists = Profile.objects.filter(id\_user=ids)

            username\_profile\_list.append(profile\_lists)

        username\_profile\_list = list(chain(\*username\_profile\_list))

    return render(request, 'search.html', {'user\_profile': user\_profile, 'username\_profile\_list': username\_profile\_list})

@login\_required(login\_url='signin')

def upload(request):

    if request.method == 'POST':

        user = request.user.username  # Lấy username của user đã đăng nhập

        image = request.FILES.get('image\_upload')  # Lấy hình ảnh từ form

        caption = request.POST['caption']  # Lấy caption từ form

        new\_post = Post.objects.create(user=user, image=image, caption=caption)  # Tạo mới Post

        new\_post.save()  # Lưu post vào cơ sở dữ liệu

        return redirect('/')  # Chuyển hướng về trang chủ sau khi đăng bài

    else:

        return redirect('/')  # Nếu không phải POST, quay lại trang chủ

@login\_required(login\_url='signin')

def like\_post(request):

    username = request.user.username

    post\_id = request.GET.get('post\_id')

    try:

        post = Post.objects.get(id=post\_id)

    except Post.DoesNotExist:

        return redirect('/')  # Redirect if the post doesn't exist

    like\_filter = LikePost.objects.filter(post\_id=post\_id, username=username).first()

    if like\_filter is None:

        new\_like = LikePost.objects.create(post\_id=post\_id, username=username)

        post.no\_of\_likes += 1

        post.save()

    else:

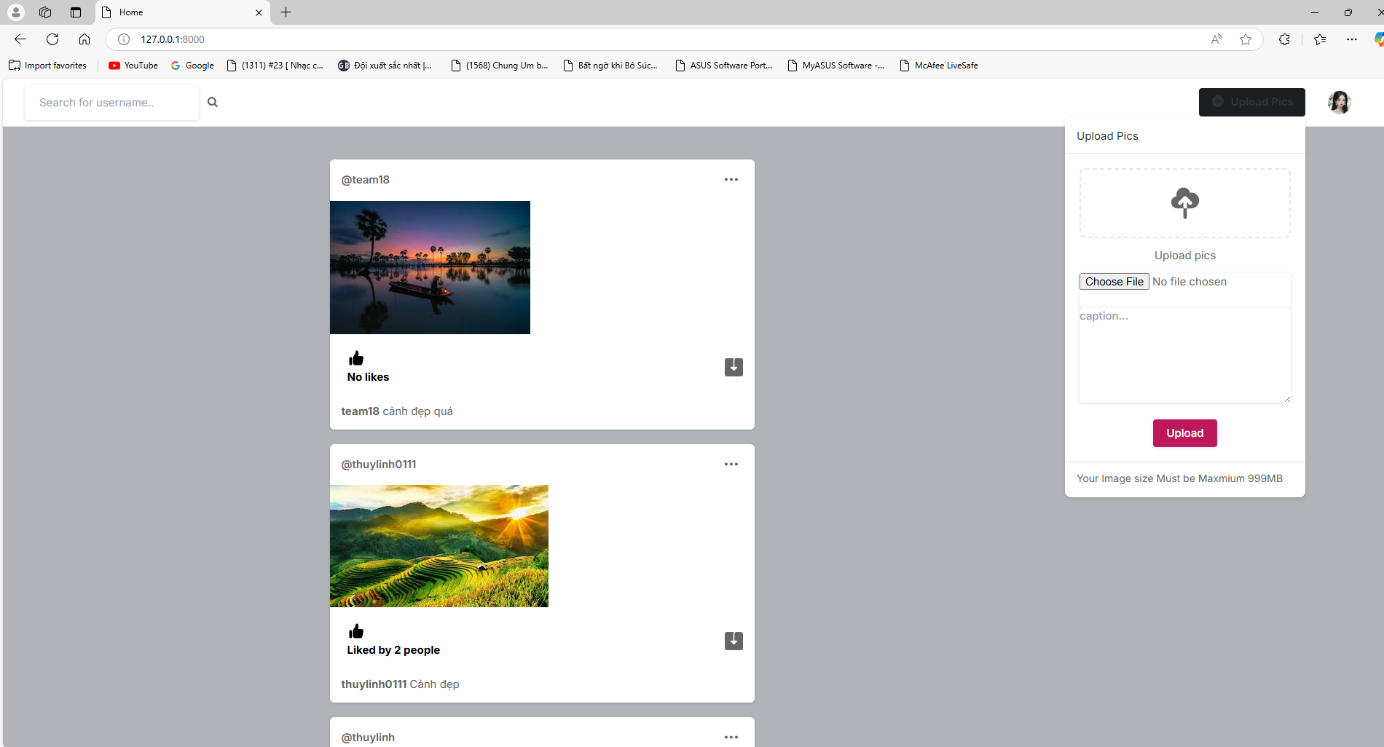
        like\_filter.delete()

        post.no\_of\_likes -= 1

        post.save()

    return redirect('/')

* Giao diện



#### Kịch bản 5: Cấu hình trang cá nhân

* Tính năng cần có

Hiện thị trang cá nhân của người dùng gồm: tên, trạng thái, đếm lượt post bài, lượt follower, lượt following và hiện thị bài viết từng post.

* Mô hình cơ sở dữ liệu

Model.py:

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE) # Liên kết một-một với User

bio = models.TextField(blank=True)

profileimg = models.ImageField(upload\_to='profile\_images', default='andanh.jpg')

location = models.CharField(max\_length=100, blank=True)

def \_\_str\_\_(self):

return self.user.username

class Post(models.Model):

id = models.UUIDField(primary\_key=True, default=uuid.uuid4, editable=False) # Sửa thành UUIDField

user = models.CharField(max\_length=100)

image = models.ImageField(upload\_to='post\_images')

caption = models.TextField()

created\_at = models.DateTimeField(default=datetime.now)

no\_of\_likes = models.IntegerField(default=0)

def \_\_str\_\_(self):

return self.user

class FollowersCount(models.Model):

follower = models.CharField(max\_length=100)

user = models.CharField(max\_length=100)

def \_\_str\_\_(self):

return self.user

* Xử lý logic đăng ký (Views)

Views.py:

@login\_required(login\_url='signin')

def profile(request, pk):

user\_object = User.objects.get(username=pk)

user\_profile = Profile.objects.get(user=user\_object)

user\_posts = Post.objects.filter(user=pk)

user\_post\_length = len(user\_posts)

follower = request.user.username

user = pk

if FollowersCount.objects.filter(follower=follower, user=user).first():

button\_text = 'Unfollow'

else:

button\_text = 'Follow'

user\_followers = len(FollowersCount.objects.filter(user=pk))

user\_following = len(FollowersCount.objects.filter(follower=pk))

context = {

'user\_object': user\_object,

'user\_profile': user\_profile,

'user\_posts': user\_posts,

'user\_post\_length': user\_post\_length,

'button\_text': button\_text,

'user\_followers': user\_followers,

'user\_following': user\_following,

}

return render(request, 'profile.html', context)

@login\_required(login\_url='signin') # Ensure the user is logged in

def follow(request):

if request.method == 'POST':

follower = request.POST['follower'] # Get the follower's username

user = request.POST['user'] # Get the user to be followed/unfollowed

# Check if the follower already follows the user

if FollowersCount.objects.filter(follower=follower, user=user).first():

# If already following, unfollow (delete the record)

delete\_follower = FollowersCount.objects.get(follower=follower, user=user)

delete\_follower.delete()

return redirect('/profile/' + user)

else:

# If not following, create a new follower record

new\_follower = FollowersCount.objects.create(follower=follower, user=user)

new\_follower.save()

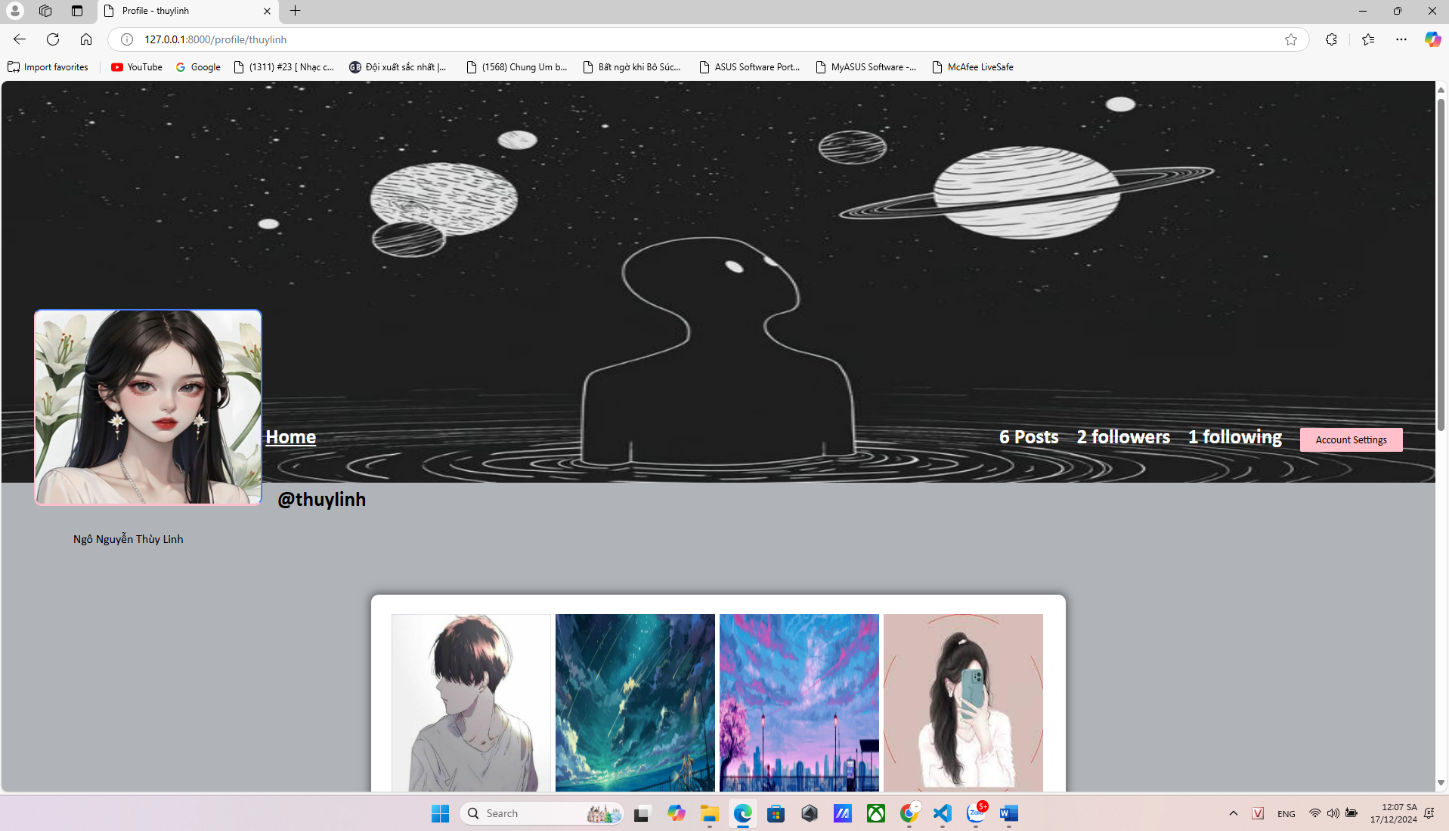
return redirect('/profile/' + user)

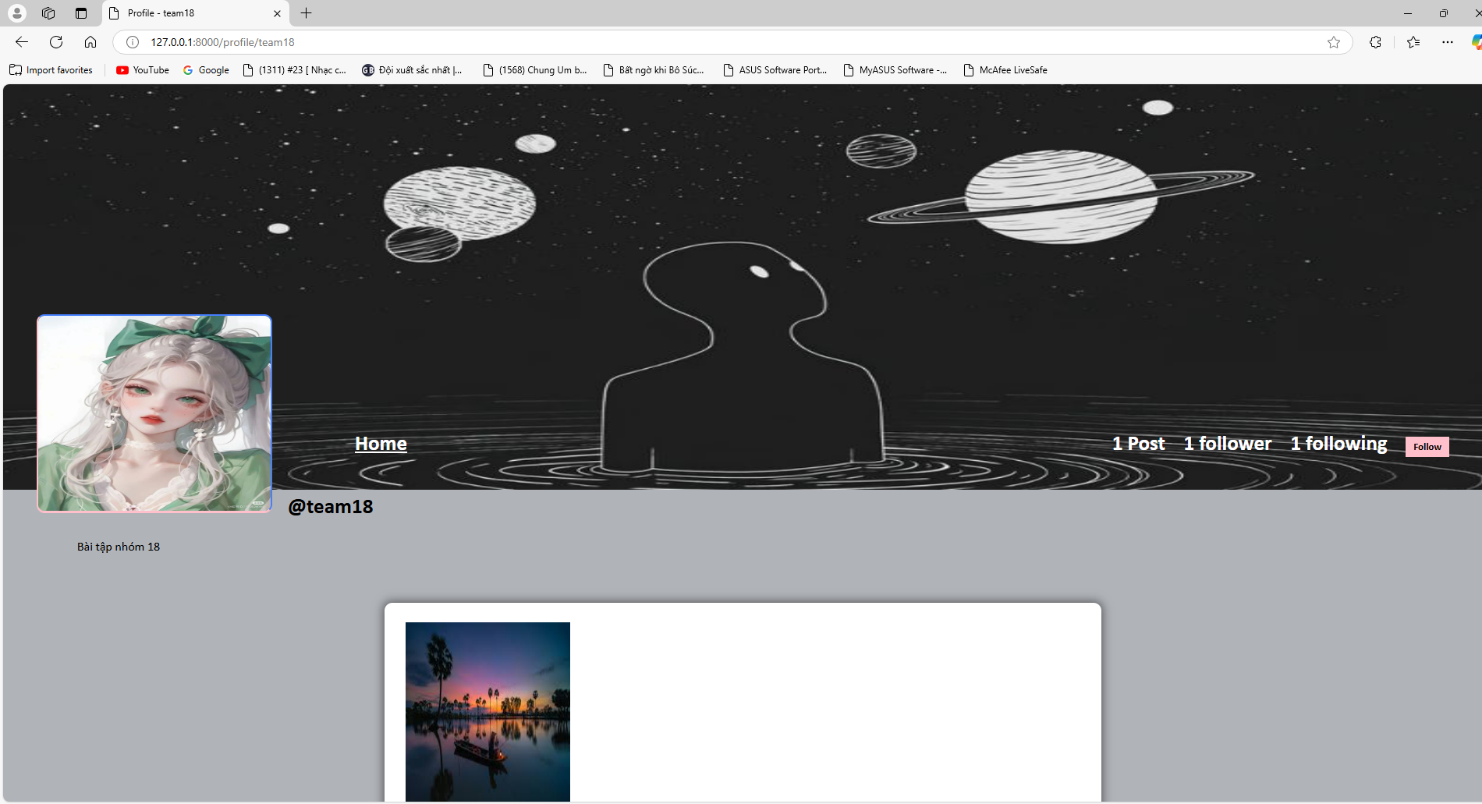
else:

# Redirect to home page if not POST

return redirect('/')

* Giao diện





### **Kiểm thử hệ thống**

#### Các loại kiểm thử

- Kiểm thử chức năng:

+ Mục đích: Đảm bảo rằng tất cả các tính năng của hệ thống hoạt động đúng như yêu cầu đã định.

+ Phương pháp: Sử dụng pytest để kiểm thử các mô hình, view, và form.

- Kiểm thử hiệu suất:

+ Mục đích: Đo lường thời gian phản hồi và khả năng xử lý đồng thời của ứng dụng.

+ Phương pháp: Sử dụng Locust để mô phỏng nhiều người dùng truy cập cùng lúc và đo lường hiệu suất.

#### Kế hoạch kiểm thử

Kế hoạch kiểm thử bao gồm các bước sau:

- Xác định các trường hợp kiểm thử: Ghi lại danh sách tất cả các tình huống cần kiểm tra cho từng chức năng hệ thống, bao gồm cả kiểm thử tự động và kiểm thử thủ công.

- Thực hiện kiểm thử: Sử dụng Postman để thực hiện các kiểm thử API, giúp xác minh rằng các điểm cuối (endpoints) của API trả về phản hồi đúng và dữ liệu hợp lệ. Kiểm thử các yêu cầu như đăng nhập, truy cập thông tin, thêm sửa ,xóa bài đăng,...

- Phân tích kết quả: Đánh giá các lỗi phát sinh và lập kế hoạch khắc phục. Tạo báo cáo kiểm thử để trình bày các phát hiện và đề xuất cải tiến cho nhóm phát triển.

#### Kết quả đạt được

Hệ thống đã được triển khai thành công và đạt được các kết quả đáng chú ý sau:

- Chức năng hoạt động ổn định: Tất cả các chức năng cơ bản của hệ thống như đăng nhập, đăng ký,thêm ,sửa ,xóa,... tìm kiếm thông tin ,một cách hiệu quả và không mắc những lỗi nghiêm trọng trong quá trình kiểm thử các chức năng của trang website mạng xã hội .

- Thời gian tải trang: Thời gian tải trang đã được tối ưu hóa và hiện đạt dưới 3 đến 4 giây, giúp cải thiện trải nghiệm người dùng và giảm tỷ lệ thoát.

- Khả năng xử lý đồng thời: Hệ thống có thể xử lý đồng thời nhiều yêu cầu từ người dùng mà không gặp phải tình trạng chậm trễ hoặc gián đoạn. Các bài kiểm thử hiệu suất cho thấy rằng hệ thống vẫn hoạt động mượt mà với khoảng 1000 người dùng truy cập đồng thời.

- Bảo mật: Sau khi kiểm thử bảo mật, không phát hiện lỗ hổng nghiêm trọng nào.

### **KẾT LUẬN**

#### Kết quả đạt được

- Hệ thống hoạt động ổn định và hiệu quả: Sau quá trình triển khai và kiểm thử, hệ thống đã cho thấy sự ổn định, với tất cả các chức năng chính hoạt động mượt mà, đáp ứng nhu cầu của người dùng mà không gặp phải sự cố lớn.

- Với chức năng thêm ,sửa ,xóa ,đăng ký và đăng nhập một cách nhanh chóng .Các chức năng được liên kết một cách chặc chẽ và đầy đủ ,hiệu quả đáp ứng được các yêu của người dùng ,cho phép người dùng đăng bài ,chia sẻ các video ,hình ảnh ,...và cho phép người dùng kinh doanh ol tại kênh đã đăng ký .

- Đảm bảo bảo mật thông tin người dùng: Hệ thống đã được thiết kế với các biện pháp bảo mật hiện đại, đảm bảo rằng thông tin quan trọng của người dùng được bảo vệ an toàn. Kiểm thử bảo mật không phát hiện lỗ hổng nghiêm trọng nào.

#### Hạn chế:

- Một số tính năng nâng cao chưa được triển khai hoàn chỉnh: Một số tính năng như phân tích dữ liệu người dùng và quản lý được hết tất cả các thông tin của người dùng có nguy cơ bị rò rỉ ra phía ngoài vẫn chưa được phát triển đầy đủ, điều này có thể hạn chế khả năng cạnh tranh của hệ thống trong tương lai.

- Cần cải thiện thêm về giao diện người dùng để dễ sử dụng hơn: Giao diện người dùng hiện tại có thể cải thiện để trở nên trực quan và dễ sử dụng hơn, giúp người dùng có trải nghiệm tốt hơn khi tương tác với hệ thống.

- Cần tiếp tục theo dõi hiệu suất để tối ưu hóa trong tương lai: Mặc dù hệ thống hiện tại hoạt động tốt, nhưng việc theo dõi hiệu suất liên tục là cần thiết để phát hiện sớm các vấn đề và tối ưu hóa hệ thống cho các nhu cầu trong tương ứng hệ thống đã được cấu hình để bảo vệ chống lại các mối đe dọa như SQL injection, XSS và CSRF. Các thông tin quan trọng của người dùng được mã hóa và bảo vệ.

- Phản hồi từ người dùng: Nhận được phản hồi tích cực từ người dùng thử nghiệm về giao diện người dùng và sự dễ dàng trong việc sử dụng. Giao diện thân thiện và dễ dàng điều hướng đã được ghi nhận.

#### Kết luận chương 3

- Chương này đã trình bàychi tiết các quy trình tiến hành kiểm thử hệ thống trang website mạng xã hội mang lại cho người dùng bằng việc sử dụng python django.Việc triển khai các hệ thống một cách hiệu quả và đầy những tính năng linh hoạt giúp trang website phong phú và đa chức năng hơn

- Kiểm thử hệ thống cho thấy được kết quả như mong đợi các tính năng hoạt động một cách chính xác và hiệu quả đạt được yêu cầu của người dùng đặt biệt là về tính năng bảo mật cao

# KẾT LUẬN

Kết quả đạt được

- Hệ thống hoạt động ổn định và hiệu quả: Sau quá trình triển khai và kiểm thử, hệ thống đã cho thấy sự ổn định, với tất cả các chức năng chính hoạt động mượt mà, đáp ứng nhu cầu của người dùng mà không gặp phải sự cố lớn.

-Với chức năng thêm ,sửa ,xóa ,đăng ký và đăng nhập một cách nhanh chóng .Các chức năng được liên kết một cách chặc chẽ và đầy đủ ,hiệu quả đáp ứng được các yêu của người dùng ,cho phép người dùng đăng bài ,chia sẻ các video ,hình ảnh ,...và cho phép người dùng kinh doanh ol tại kênh đã đăng ký .

- Đảm bảo bảo mật thông tin người dùng: Hệ thống đã được thiết kế với các biện pháp bảo mật hiện đại, đảm bảo rằng thông tin quan trọng của người dùng được bảo vệ an toàn. Kiểm thử bảo mật không phát hiện lỗ hổng nghiêm trọng nào.

Hạn chế:

- Một số tính năng nâng cao chưa được triển khai hoàn chỉnh: Một số tính năng như phân tích dữ liệu người dùng và quản lý được hết tất cả các thông tin của người dùng có nguy cơ bị rò rỉ ra phía ngoài vẫn chưa được phát triển đầy đủ, điều này có thể hạn chế khả năng cạnh tranh của hệ thống trong tương lai.

- Cần cải thiện thêm về giao diện người dùng để dễ sử dụng hơn: Giao diện người dùng hiện tại có thể cải thiện để trở nên trực quan và dễ sử dụng hơn, giúp người dùng có trải nghiệm tốt hơn khi tương tác với hệ thống.

- Cần tiếp tục theo dõi hiệu suất để tối ưu hóa trong tương lai: Mặc dù hệ thống hiện tại hoạt động tốt, nhưng việc theo dõi hiệu suất liên tục là cần thiết để phát hiện sớm các vấn đề và tối ưu hóa hệ thống cho các nhu cầu trong tương lai

Những Điểm Nổi Bật

Tính năng chính:

Post bài viết: Người dùng có thể đăng bài viết và hiển thị chúng trên trang cá nhân hoặc trang chung.

Like bài viết: Cho phép người dùng tương tác bằng cách thích bài viết.

Follow người dùng: Tạo mối quan hệ giữa các người dùng, giúp hiển thị bài viết từ người được theo dõi.

Sử dụng Django Framework:

Django hỗ trợ nhanh chóng xây dựng các chức năng này nhờ hệ thống ORM mạnh mẽ, hệ thống xác thực tích hợp, và cơ chế quản lý URL linh hoạt.

Việc phát triển ứng dụng đảm bảo an toàn, dễ mở rộng và thân thiện với lập trình viên.

# TÀI LIỆU THAM KHẢO

Tài liệu tiếng việt và tiếng anh

1. <https://docs.djangoproject.com/en/5.1/>

2. <https://github.com/pavan6029/django-social-media>

###### 3. <https://simpleisbetterthancomplex.com/>

4. <https://dev.to/>

5. <https://getbootstrap.com/docs/5.3/getting-started/download/>

6. <https://www.youtube.com/watch?v=hyQq7RJkGB8&list=PLUocOGc7RDEIChluBEd5bxuU8Z86-0PyO>

7. <https://www.youtube.com/watch?v=BGEJsk6kSoI&list=PLUocOGc7RDEIChluBEd5bxuU8Z86-0PyO&index=2>

8. <https://www.youtube.com/watch?v=_qOgy8R8PJQ&list=PLUocOGc7RDEIChluBEd5bxuU8Z86-0PyO&index=3>

9. <https://www.youtube.com/watch?v=L55V6wYEPVA&list=PLUocOGc7RDEIChluBEd5bxuU8Z86-0PyO&index=4>

10. <https://www.python.org/>

11. <https://www.djangoproject.com/>

12. <https://pypi.org/project/pillow/>

13. <https://www.mysql.com/downloads/>

14. <https://django-crispy-forms.readthedocs.io/en/latest/install.html>

Mẫu 2

**ĐẠI HỌC DUY TÂN**

TRƯỜNG KHOA HỌC MÁY TÍNH

**CS466 – Perl & Python Bảng điểm đánh giá Đồ án nhóm**

Tên đề tài: LẬP TRÌNH WEBSITE MẠNG XÃ HỘI BẰNG DJANGO TRONG PYTHON

Hướng dẫn/Phản

biện: Th.S Trần Hữu Minh Đăng

Tên sinh viên 1: Ngô Nguyễn Thùy Linh(sv1) ID: 28201139984 Tên sinh viên 2: Trần Thị Thanh Giang (sv2) ID: 28201154741

|  |  |  |  |
| --- | --- | --- | --- |
| **Đầu ra** | **Tiêu chí** | **Điểm** | **Phần trăm** |
| Thiết kế (25%) | Định nghĩa rõ các yêu cầu và ý tưởng triển khai, cài đặt. | **S1:15%** | 25% |
| **S2: 10%** |
| Triển Khai (25%) | Đánh giá giải pháp thiết kế so với các yêu cầu cho trước | **S1:18%** | 25% |
| **S2:7%** |
| Báo cáo viết (25%) | Bảng tiêu chí đánh giá giao tiếp viết | **S1:8%** | 25% |
| **S2:17%** |
| Kĩ năng nhóm (25%) | Bảng tiêu chí đánh giá Kĩ năng nhóm | **S1:12,5%** | 25% |
| **S2:12,5%** |
|  | **Điểm tổng:** | **S1:53,5%** | 100% |
| **S2:46,5%** |

**Góp ý, đánh giá, bình luận:**

Ngày :18/12/2024

**Chữ kí người Hướng dẫn/Phản biện**

**ĐẠI HỌC DUY TÂN**

**Trường Khoa học Máy tính**

**Bảng tiêu chí đánh giá kĩ năng giao tiếp viết**

Môn học: CS 466 E Học kì/Năm học: 1/2024-2025

Mẫu 3

Tên giảng viên: Th.S Trần Hữu Minh Đăng

Tên dự án: LẬP TRÌNH WEBSITE MẠNG XÃ HỘI BẰNG DJANGO TRONG PYTHON

Tên (các) sinh viên: Ngô Nguyễn Thùy Linh, Trần Thị Thanh Giang

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **1 – Không thành thạo** | **2 – Ít thành thạo** | **3 – Thành thạo mức trung bình** | **4 – Thành thạo** | **5 – Rất thành thạo** | **Điểm** |
| Cấu trúc bài viết | Không có tổ chức hoặc luồng thông tin hợp lý | Nội dung không phù hợp của hầu hết các phần trong báo cáo | Không quá 2 trường hợp nội dung nằm trong phần không phù hợp của báo cáo | Nội dung phù hợp với tất cả các phần của báo cáo | Tổ chức/bố cục nâng cao khả năng hiểu tác phẩm của người đọc, đồng thời vẫn tuân theo các hướng dẫn | Sv1/4  Sv2/4 |
| Mức độ hoàn thành bài viết | Không có câu chuyện nào được kể, rất không đầy đủ | Thiếu các khía cạnh của toàn bộ câu chuyện | Các khía cạnh nhỏ của câu chuyện không được đưa vào | Câu chuyện được kể đầy đủ | Tài liệu bổ sung nâng cao chất lượng báo cáo | Sv1/3    Sv2/3 |
| Thẩm mỹ | Không thể đọc/hiểu các bảng và hình ảnh, phông chữ không rõ ràng | Văn bản, bảng và hình ảnh khó hiểu và/hoặc khó đọc | Các phần cẩu thả và khó đọc | Văn bản, bảng biểu, hình ảnh có thể đọc được | Văn bản, bảng biểu, hình ảnh dễ đọc và dễ hiểu rõ ràng | Sv1/3    Sv2/3 |
| Định dạng | Quá nhiều lỗi định dạng khiến báo cáo trở nên vô dụng | Nhiều lỗi định dạng khiến báo cáo khó đọc | Không quá 2 lỗi định dạng | Định dạng được tuân theo | Các khía cạnh định dạng nâng cao tác động của báo cáo | Sv1/4    Sv2/4 |
| Chính tả | Nhiều lỗi chính tả | Nhiều lỗi chính tả | Không quá 2 lỗi chính tả | Không quá 1 lỗi chính tả | Không có lỗi chính tả | Sv1/2  Sv2/2 |
| Ngữ pháp và chấm câu | Lỗi ngữ pháp và dấu câu khiến báo cáo không thể đọc được | Nhiều lỗi ngữ pháp và dấu câu | Lỗi ngữ pháp và dấu câu nhỏ | Ngữ pháp đúng và không quá 2 lỗi dấu câu | Không có lỗi ngữ pháp hoặc dấu câu | Sv1/4  Sv2/4 |
| Tổng: | | | | | | Sv1 20/30  Sv2 20/30 |
| Điểm tổng trên thang 10 :  (\*10/30) | | | | | |  |

**ĐẠI HỌC DUY TÂN**

Mẫu 4

**Trường Khoa học Máy tính Bảng tiêu chí đánh giá kĩ năng làm việc nhóm**

Môn học: CS 466 E Học kì/Năm học: 1/2024-2025

Tên giảng viên: Th.S Trần Hữu Minh Đăng

Tên dự án: LẬP TRÌNH WEBSITE MẠNG XÃ HỘI BẰNG DJANGO TRONG PYTHON

Tên (các) sinh viên: Ngô Nguyễn Thùy Linh, Trần Thị Thanh Giang

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Thuộc tính** | **1 – Không thành thạo** | **2 – Ít thành thạo** | **3 – Thành thạo mức trung bình** | **4 – Thành thạo** | **5 – Rất thành thạo** | **Điểm** |
| **Đóng góp cho dự án/công việc của nhóm** | Không tham gia vào hoạt động làm việc nhóm | Không thu thập bất kỳ thông tin có liên quan nào; không có gợi ý hữu ích nào để giải quyết nhu cầu của nhóm | Thu thập thông tin khi được thúc đẩy; cố gắng đưa ra một số ý tưởng, nhưng không  được phát triển tốt và không được diễn đạt rõ ràng, để  đáp ứng nhu cầu của nhóm | Thu thập thông tin cơ bản, hữu ích liên quan đến dự án; thỉnh thoảng đưa ra những ý tưởng hữu ích để đáp ứng nhu cầu của nhóm | Thu thập và trình bày cho nhóm rất nhiều thông tin có liên quan; đưa ra những ý tưởng được phát triển tốt và diễn đạt rõ ràng liên  quan trực tiếp đến mục đích của nhóm | Sv1/4  Sv2/3 |
| **Chịu trách nhiệm** | Không chịu trách nhiệm cho công việc nhóm hoặc công việc của riêng mình | Không thực hiện nhiệm vụ được giao; thường xuyên vắng mặt trong các cuộc họp và khi có mặt, không có điều  gì mang tính xây dựng để nói; dựa vào người khác để làm  việc | Thực hiện nhiệm vụ được giao nhưng cần nhiều lời nhắc nhở; tham dự các cuộc họp thường xuyên nhưng nhìn chung không nói điều gì mang tính xây dựng; đôi khi mong đợi người khác  làm công việc của mình | Thực hiện tất cả các nhiệm vụ được giao; tham dự các cuộc họp thường xuyên và thường tham gia hiệu quả; nhìn chung đáng tin cậy | Thực hiện tất cả các nhiệm vụ rất hiệu quả; tham dự tất cả các cuộc họp và tham gia nhiệt tình; rất đáng tin cậy | Sv1/4  Sv2/4 |
| **Các vấn đề phụ dành cho thành viên nhóm** | Các vấn đề phụ không được định nghĩa hoặc được định nghĩa rất mơ hồ. Các thành viên nhóm bối rối và Có sự không phù hợp lớn với các mục tiêu chung của dự án. | Các vấn đề phụ được định nghĩa không rõ ràng. Có một số sự không phù hợp với các mục tiêu chung của dự án. Có các nhiệm vụ trùng lặp. | Các vấn đề phụ được định nghĩa với độ rõ ràng vừa phải. Một số thành viên nhóm có thể có câu hỏi hoặc sự không chắc chắn về  nhiệm vụ và mục tiêu của họ. | Các vấn đề phụ được định nghĩa rõ ràng và chủ yếu phù hợp với các mục tiêu chung của dự án. Hầu hết các thành viên nhóm đều hiểu nhiệm vụ và mục tiêu của họ. | Các vấn đề phụ được định nghĩa với độ rõ ràng. Các thành viên nhóm hiểu chính xác về nhiệm vụ, mục tiêu và thời hạn của họ. Tất cả các vấn đề phụ đều phù hợp với  các mục tiêu chung của dự án | Sv1/3  Sv2/3 |
| **Mức độ hoàn thiện của giải pháp cuối cùng.** | Giải pháp cuối cùng không hoàn thiện, giải quyết hầu như không có khía cạnh nào của vấn đề. | Giải pháp cuối cùng không hoàn thiện, giải quyết một số khía cạnh của vấn đề. Nó bao  gồm một số giải pháp phụ cần thiết. | Giải pháp cuối cùng khá hoàn thiện, giải quyết một số khía cạnh của vấn đề. Nó  bao gồm hầu hết các giải pháp phụ. | Giải pháp cuối cùng hầu như hoàn thiện, giải quyết phần lớn các khía cạnh của vấn đề, mặc dù nó có thể bao gồm tất cả  các giải pháp phụ. | Giải pháp cuối cùng là hoàn thiện, giải quyết tất cả các khía cạnh của vấn đề. Nó bao gồm tất cả các giải pháp phụ. | Sv1/3  Sv2/3 |
| **Phần mềm quản lý dự án** | Không sử dụng phần mềm | Phần mềm được sử dụng,  nhưng chỉ cho một trong ba tính năng: lập kế hoạch, theo dõi và báo cáo. | Phần mềm được sử dụng,  nhưng chỉ cho hai trong ba tính năng: lập kế hoạch, theo dõi và báo cáo. | Phần mềm được sử dụng, cho  tất cả ba tính năng: lập kế hoạch, theo dõi và báo cáo, nhưng không tối ưu lắm | Phần mềm được sử dụng, cho tất  cả ba tính năng: lập kế hoạch, theo dõi và báo cáo, giúp tối ưu hóa tốc độ dự án. | Sv1/4  Sv2/4 |
|  | | | | | |  |
| Tổng điểm: | | | | | | Sv1 18/25  Sv2 17/25 |
| Tổng điểm theo thang 10 :  (\*10/25) | | | | | |  |